23.3.90

LOOP

24

März 1990

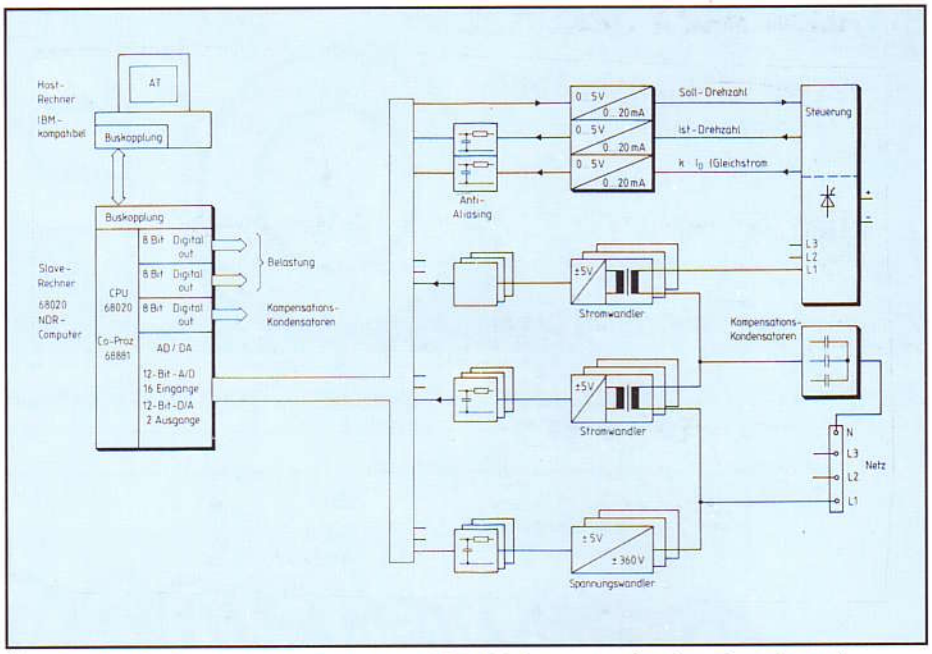
Zeitung für Computer-Bauer, -Anwender, -Programmierer und -Starter

DM 3,50

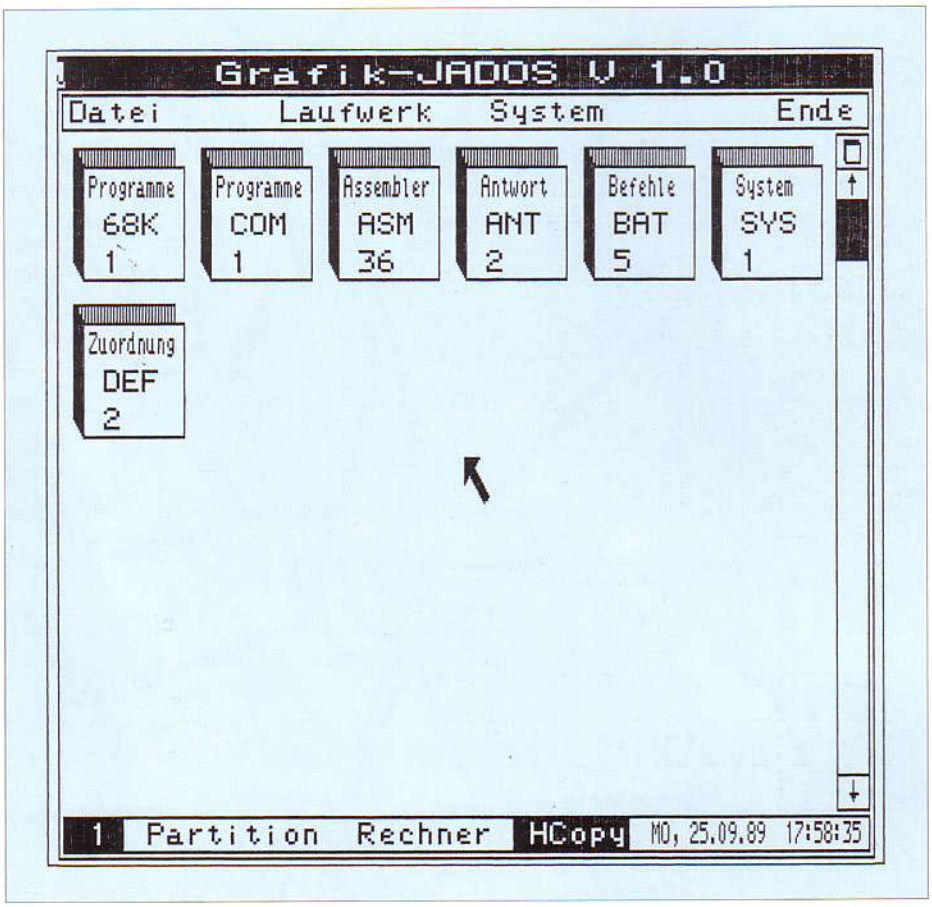
Leitartikel

CPU Z80

CPU 680XX



Übersicht der Rechnerkonfiguration (GRAF-Computer in der Forschung)



Grafik-JADOS

mc-modular-AT

CPU 8088 - MS-DOS

In eigener Sache

Rubriken

Impressum

LOOP Zeitung für Computerbauer

Herausgeber: Gerd Graf

Redaktion: Rolf Dieter Klein, Gerd Graf, Ulrich Kracker, Nikolaus Bischof

Gestaltung: bbs Computer Systeme, Elisabeth Mayr, MARTEN

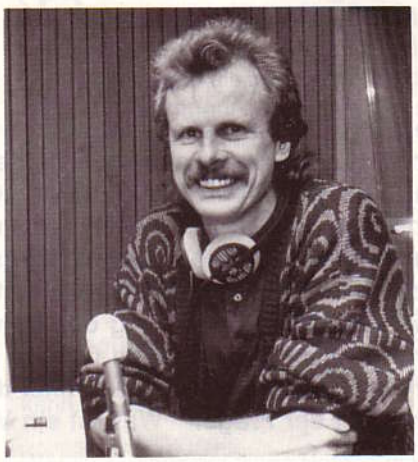
Druck: Karl-Heinz-Rieder, Kempten

Herstellung und Anzeigenverwaltung: GES GmbH, Magnusstr. 13, 8960 Kempten

Anzeigenpreisliste: 02/90

Editorial

Loop regelmäßig



Schon oft be- oder geschworen, wird es so langsam Tatsache - die LOOP erscheint regelmäßig!

Alle, die eine Zeitung machen - sei es eine Vereinszeitung, eine Klassenzeitschrift oder eine "richtige" Zeitung, können uns sicher nachfühlen, daß es nicht leicht ist, eine regelmäßige Zeitschrift auf einem hohen Niveau immer rechtzeitig erscheinen zu lassen.

Die Schwerpunkt- und Erscheinungsthemen der LOOP für 1990 liegen bereits fest und einige Artikel für die nächste LOOP liegen bereits druckfertig vor - lassen Sie sich überraschen!

Für die Zukunft wollen wir die LOOP noch interessanter für Sie gestalten. Sie wird mehr konkrete Artikel und Anwendungen für Anfänger enthalten, mehr Grundlagenartikel, wie wir hier schon über DSP zeigen, und mehr Beschreibungen. Natürlich wollen wir nicht ohne Sie planen - deshalb:

Ihre Meinung über LOOP

Schreiben Sie uns Ihre Meinung über die LOOP. Was Sie lesen wollen. Was Sie nicht lesen wollen. Was Ihnen gefällt. Was nicht. Einfach alles. Wir wollen IHRE Zeitung machen.

Schreiben Sie auch wieder Leserbriefe. In einem Leserbrief kann man wunderbar über Produkte schimpfen, Fragen stellen, Tips und Tricks einholen oder verraten, Kontakte knüpfen und vieles mehr. Nur zu!

NDR-Produkte - was noch?

So langsam sind wir am Ende mit den Ideen zum NDR-Computer. Oder haben wir eine Baugruppe vergessen, auf die vielleicht gerade SIE warten ? Schreiben Sie uns!

Der neue Prommer

In dieser LOOP stellen wir die neue Prommer-Baugruppe vor. Der "alte" Promer war ja einer der ersten Baugruppen für den NDR-Computer und wurde nie erweitert - deshalb nun ein völlig neues Gerät, das alle Wünsche an einen universellen EPROM-Programmierer erfüllen sollte.

Neu am Promer ist, daß dieser auch mit einer PC-Bus-Belegung erhältlich ist, und dies verbunden mit einer ganz ausgezeichneten Software. Die PC-Promer-SW ist im neuen IBM-SAA-Standard geschrieben, natürlich mit Window-Technologie, Maus-Bedienung und vieles mehr. Der Autor hat versprochen, in den nächsten LOOPs einiges über diese Art der Programmierung zu verraten. Seien Sie also gespannt!

CeBit 90 - willkommen

Wohnen Sie in der Nähe Hannovers oder müssen / dürfen Sie beruflich zur CeBit - besuchen Sie uns! Sie finden uns dieses Jahr in einer neuen Halle - der Halle 7, gleich links vom Eingang. Microsoft ist gegenüber. Die (sehr schmale) LOOP-Redaktion, bestehend aus Herrn Bischof, Herrn Kracker und mir wird vollständig vertreten sein und freut sich über Ihre Wünsche oder Kritik.

Bis zur CeBit oder bis zu Ihrem nächsten Brief

Gerd Graf

Freitag, der 13te

Teil 1 - Was ist ein Virus?

Wenn man den Pressemeldungen Anfang Oktober 1989 Glauben schenken wollte, so mußte man sich auf den Untergang der EDV-Welt vorbereiten. Die Viren ISRAELI #1 und DATACRIME würden jedem Computer, der von ihnen befallen sei, die Bits austreiben. Da der Laie von Computerviren nur ein sehr biologisch geprägtes Bild hat, die Boulevard-Presse dies auch noch zu reisserischen Sensationsmeldungen über die drohende (physikalische) Zerstörung der Computer mißbraucht, sollte für Leute vom Fach, und damit auch für uns LOOP-Leser, Aufklärung das Gebot der Stunde sein. Der erste Artikel dieser dreiteiligen Reihe befaßt sich daher mit der Frage: " Was ist ein Virus ? ".

Eine gute und kurze Antwort findet man in [4]:"...Ein Programm mit zerstörerrischen Auftrag (Schadensprogramm) wird in einem anderen Programm versteckt, und entfaltet seine unheilvolle Wirkung zeitversetzt. Gleichzeitig nutzt das Schadensprogramm die Zeit bis sich in möglichst viele andere Programme zu kopieren. Der Mensch mit seiner Fähigkeit zum Disketentransport, oder auch die Vernetzung von Rechnern, helfen dem Schadensprogramm die Brücke zu anderen Computern zu schlagen. Wegen der Verhaltensanalogie zu biologischen Viren, werden solche Schadensprogramme oft auch als 'ComputerViren' bezeichnet."
Um den doch großen Unterschied zu verdeutlichen, werde ich aber "der Computer-Virus" im Gegensatz zu "das biologische Virus" schreiben.

Außerdem möchte ich mich auf die sog. "stand-alone"-Rechner beschränken und die Probleme vernetzter oder großer Rechenanlagen nicht mit erörtern. Aber auch so gibt es bei Computer-Viren einen Artenreichtum, der bereits heute erschreckend groß ist.

Kurze Viren-Geschichte

Das offizielle Geburtsjahr des ersten Virus ist 1983. Damals beschrieb der Amerikaner Fred Cohen im Rahmen einer Dissertation [2] die Funktionsweise von Viren und demonstrierte die Durchseuchung an einer (damals als sicher geltenden) VAX unter dem Betriebssystem UNIX. Aber erst durch seinen Vortrag auf dem 7. nationalen Kongreß über Computer-Sicherheit 1984 erlangte das Thema "Viren" die verdiente Beachtung [3]. Ein Beispiel für die rasante Verbreitung ist der ISRAELI #1. Zuerst im November 1987 in Haifa, dann in Tel Aviv festgestellt, tauchte der Virus im Dezember 1987 in Jerusalem auf und war im Frühjahr '88 schon weltweit zu finden.

Große Bekanntheit erlangte 1988 hierzulande der sog. "c't-Virus" durch einen Artikel komplett mit Quell- und entsprechendem Anti-Virus-Programm im gleichnamigen Magazin. Diese spektakuläre Veröffentlichung wurde allerdings von der Fachwelt einhellig und zu Recht verurteilt. Den bisherigen Höhepunkt in der Diskussion stellt nun der Oktober '89 dar. IBM rang sich sogar zu einem Virus-Alarm durch und verteilte Anti-Viren-Programme, die leider viele Benutzer zu spät, d.h. nach dem gefürchteten Freitag, dem 13., erreichte. Aber es war ein Schritt in die richtige Richtung. Bisher erklärten sich die Software-Produzenten für die Probleme ihrer Kunden mit Viren nicht zuständig (vielleicht war für diesen Sinneswandel ausschlaggebend, daß eine große Firma mehrere Monate lang eine mit einem Virus verseuchte Mutter-Diskette hatte, von der die Kopien für die Kunden gemacht wurden).

Aufbau eines Virus

```
PROGRAM Virus_mit_Auslöser;
...
    PROCEDURE Infiziere;
    (* infiziere Datei nach Strategie aus (A)-(C) *)
    ...
    PROCEDURE Funktion;
    (* führe eine bestimmte Funktion aus *)
    ...
    PROCEDURE Auslöser;
    (* teste ob Auslösebedingung gegeben ist *)
...
BEGIN
    Infiziere;
    IF Auslöser
    THEN Funktion
END.
```

Bild 1: Ein Virus nistet sich ein.

Grundsätzlich muß jeder Virus eine Funktions- und eine Infektionsroutine haben. Fast immer ist aber noch ein Auslöser zu finden, der auf ein bestimmtes Ereignis hin die Funktionsroutine aufruft (Bild 1).

Die Infektionsroutine ist für die Vermehrung des Virus zuständig. Das Hauptzieleines jeden Virus ist es, sich in eine ausführbare Datei einzupflanzen, da Viren als unselbständige Programmstücke sich nicht selbst ausbreiten können. Dieser Vorgang ist i.a. erstaunlich kurz und bewegt sich in einem Zeitrahmen von ca. 1/2 bis 5 sec.
Die Kriterien für den Aufbau der Infektionsroutine lassen sich in drei Gruppen aufteilen:

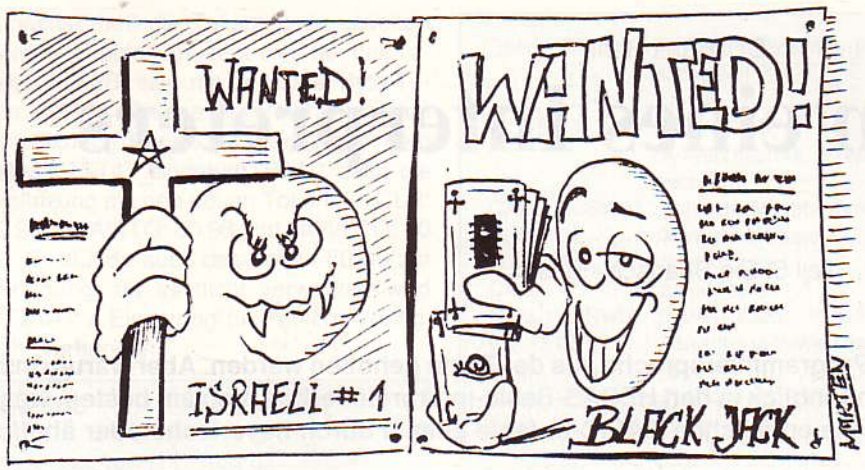
(A) Wie wird infiziert:
Überschreiben der Datei / Einfügen in Datei

(B) Wie oft wird infiz.:
einmalig, d.h. Prüfung auf Infektion / mehrmalig, d.h. ohne Prüfung auf Infektion

(C) Was wird infiziert:
Bootspuren / spezielle Dateien (z.B. *.EXE, *.68k) / alle erreichbaren Dateien

Alle Kombinationen von Konfigurationsmöglichkeiten aus (A), (B), und (C) sind denkbar.

Zu (A):
Das simple Überschreiben von Dateien ist selten, weil "unelegant". Bei dieser Methode kopiert sich der Virus über den Anfang der Datei (die zugleich dabei zerstört wird). "Vorteil" der Methode: Äußerlich ist nichts feststellbar, da die Länge der Datei gleich bleibt. "Nachteil": Der Virus ist recht kurzlebig, da die Funktionalität der verseuchten Datei nicht aufrecht erhalten werden kann. Ein nicht funktionierendes Programm fällt rasch auf und wird schnell gelöscht bzw.

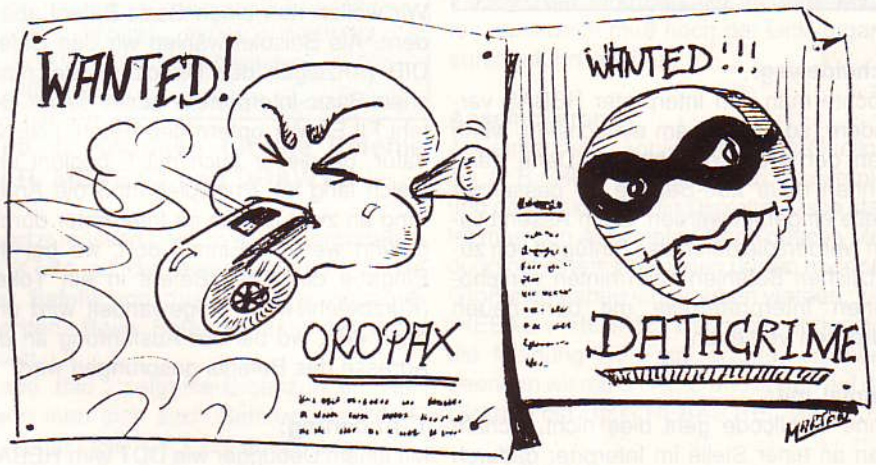


(korrekt) überschrieben.
Genau gegenteilig wirkt das Einfügen in eine Datei. Die Funktionalität einer ausführbaren Datei wird größtenteils erhalten, indem der Virus zuerst die Originaldaten ans Ende der Datei kopiert und sich selber dann vorne installiert. Im Programm auftretende Sprünge werden so umgelenkt, daß das Programm läuft, aber auch der Virus aktiviert wird. Äußerlich ist eine Änderung in der Länge der Datei erkennbar, wenn wir mal von der Möglichkeit absehen, daß genügend große Teile der Daten ausgelagert werden können, doch ist dem Benutzer die Kontrolle auf Korrektheit der Längenangaben nur selten möglich. Wer kann sich so viele Zahlen denn schon merken ?

Zu (B):
Mit dem einmaligen Infizieren ist die einfache Infektion mit nur einem Virus-Typ gemeint. Natürlich kann ein Programm auch mit den verschiedensten Viren gleichzeitig verseucht sein.

Zu (C):
Nicht nur ausführbare Dateien sind für Viren interessant. Es gibt auch Datenbank-Viren.

Kommen wir nun zur Funktionsroutine. Damit ist der Teil des Virus gemeint, der die Störung bzw. Schädigung verursacht. Dazu möchte ich einen Ausschnitt aus [1] zitieren: "...Schäden eines Computer-Virus können dauerhaft (=permanent) oder vorübergehend (=transient) sein. Bei permanenten Schäden werden etwa Boot-



sektoren überschrieben, Disketten oder Platten ganz oder teilweise gelöscht,oder es werden Sektoren (gezielt oder zufällig) als 'defekt' (Bad Sectors in der FAT = File Allocation Table) gekennzeichnet. Dagegen sind Bildschirmspiele (etwa das Herabfallen von Buchstaben in den Herbstmonaten beim Konstanzer 'Herbstlaub-Virus'), das wiederholte Abspielen von Melodien (Oropax-Virus) oder blinkende Lämpchen an Disketten und Platten nur 'vorübergehende' (wenn auch recht schockierende) Schäden, da deren Wirkung auf bestimmte Puffer (z.B. den Bildschirmpuffer) beschränkt bleiben.

Schließlich gibt es auch Viren, die normalerweise nur vorübergehende Schäden, unter besonderen Bedingungen jedoch auch permanente Schäden bewirken (transient/permanent)." Meistens ist die Schädigung noch an einen Auslöser gekoppelt. Dieser kann aktiv sein und auf eine bestimmte Eingabe von außen warten, oder er kann passiv sein und die interne Systemzeit und Datum vergleichen (-> Freitag, der 13.). "...Neben dem Erreichen bestimmter Werte eines Infektionszählers können auch Zufallswerte (zur Erschwerung der Erkennung) eine Rolle spielen..."[1].

Abschließend zu Teil 1 möchte ich alle bitten, die sich jetzt "inspiriert" fühlen und einen Virus schreiben wollen, dieser Versuchung zu widerstehen. Es wäre wichtiger, daß man die dazu nötige Energie in Überlegungen investiert, wie man die Computer-Systeme sicherer machen kann.

Wer sich unbedingt einen Namen machen will, das Gebiet des Daten-SCHUTZES bietet dazu reichlich Gelegenheit. Daß die absichtliche Verbreitung von Viren eine Straftat im Sinne von 303 b (Computersabotage) des StGB ist und kein Kavaliersdelikt, sollte wohl jedem klar sein.

Literatur:

[1] Brunnstein, Klaus Universität Hamburg, "Zur Klassifikation von Computer Viren: Der 'Computer Virus Katalog' ",Proceedings der 19. GI-Jahrestagung 1989

[2] Cohen, Fred Univerity of Southern California "Computer Viruses", Ph.D. Dissertation 1983

[3] Cohen, Fred University of Southern California: "Computer Viruses - Theory and Experiments", Proceedings of the 7th National Computer Security Conference 1984

[4] Krause, Georg INFOSYS GmbH Bodenheim "INFOCHECK" Begleitheft zum Public-Domain-Programm INFOCHECK 1989

Dr. Hans Hehl

Ökosystem eines Interpreters

Teil 5: Ein Boarisch - Basic

Es soll hier keineswegs eine bayerische Programmiersprache aus der Taufe gehoben werden. Aber warum muß es eigentlich immer Englisch sein? Einen Einblick in den HEBAS-Basic-Interpreter erhält man am besten, wenn man ihn verändert. Und so ersetzen wir die englischen BASIC-Befehle einmal durch bayerische oder ähnlich lautende.

Ein kleines Basic-Programm könnte dann in gemäßigter Form (für Norddeutsche) so aussehen:

```
10 SAGE  " GRÜASS GOTT"
20 ZEIGE "A:*.*"
30  VON I = 1 BIS 10
40     EINGABE "AUF GEHTS";X$
50  NOCHMAL
60 SERVUS
```

Scheideweg:
Möchte man den Interpreter HEBAS verändern, so geht es am einfachsten, wenn man den Quellcode besitzt. Denn dann können neue Z80-Befehle an passender Stelle eingefügt werden. Beim Assemblieren werden die durch das Einfügen von zusätzlichen Befehlen nach hinten verschobenen Interpreterteile mit den neuen Adressen versehen.

Einmal mit:
Ohne Quellcode geht dies nicht. Schafft man an einer Stelle im Interprter dadurch Platz, daß man mit einem Debugger einen Interpreterteil nach hinten verschiebt, so stimmen die weiteren Adressen und Sprungbefehle nicht mehr.

Einmal ohne:
Doch auch hier gibt es eine, wenn auch etwas umständlichere Methode. Man muß an zentraler Stelle die Adresse eines Sprungbefehls so ändern, daß dieser nun ans Ende des Interpreters oder zu einem freien Speicherbereich führt.
Dort befindet sich nun der neue Programmteil und am Ende steht ein Sprungbefehl zu der ursprünglich vorhandenen Adresse. Möglich wäre so etwas bei einem unbedingten Sprungbefehl wie z.B. JP ADRESSE beim Z80 oder bei einem Unterprogrammaufruf (CALL ADRESSE). In beiden Fällen muß man aufpassen, daß die Reihenfolge der auf dem sog. Stapel (Stack oder Kellerspeicher) abgelegten Adressen durch den Rücksprung nicht verändert wurde. Werden nur die Namen der Basic-Befehle verändert, so geht dies auch ohne Quellcode. Man muß allerdings darauf achten, daß die neuen Befehle insgesamt nicht mehr Bytes enthalten als vorher. Wer nur den EPROM-Interpreter EHEBAS besitzt, kann diesmal nur den Artikel lesen.

Frisch gewagt:
Wir wollen nun einen Basic-Befehl abändern. Als Beispiel wählen wir den Befehl DIR (Anzeigen des Directory). Bei manchen Basic-Interpretern lautet dieser Befehl FILES.Wir opfern den Befehl FNEND dafür, da dieser auch mit F beginnt und gleich lang ist. Prinzipiell muß die Änderung an zwei Stellen im Interpreter durchgeführt werden. Einmal dort, wo bei der Eingabe der Basic-Befehl in ein Token (Kurzbefehlswort) umgewandelt wird und dann dort, wo bei der Ausführung an die Adresse des Befehls gesprungen wird.

1. Änderung:
Mit einem Debugger wie DDT wird HEBAS geladen und in der Befehlstabelle ab Adresse 19DFh (Hebas-Vers. 3.1) der Befehl FNEND aufgesucht. Allerdings finden wir nur die Buchstaben NEND (bei Adresse 1A95h). Erinnern Sie sich (siehe Loop 23) an die Kennzeichnung der Basic-Befehle mit gleichem Anfangsbuchstaben durch ein gesetztes Bit 7 beim ersten Buchstaben des ersten Befehls? Die Bytes 4E 45 4E 44 ersetzen wir durch 49 4C 45 53, also den Text NEND durch ILES.

2. Änderung:
Nun müssen wir erreichen, daß der Interpreter bei der Programmausführung den FILES-Befehl als DIR-Befehl ausführt. Dazu muß in der ab 14Fh beginnenden Sprungtabelle die Adresse des FNEND-Befehls durch die des DIR-Befehls ersetzt werden. Ab Adresse 15Fh ersetzen wir die Bytes 0E 0A durch 74 35. Die neue Adresse lautet also 3574h und nicht mehr 0A0Eh.

Fertig:
Nach dem Verlassen des Debuggers und dem beim DDT notwendigen Abspeichern des Programmes mit "SAVE 73 TEST.COM" funktionieren der Befehl FILES und DIR. Allerdings ist dies nicht sehr vorteilhaft. Besser geht es, wenn der Quellcode zur Verfügung steht.
den.

Neue Basic-Befehle:
So kann man z.B. Graphik-Befehle unter Ausnützung der schnellen FLOMON-Graphik-Einsprünge beim NDR-Rechner erstellen. Beispielhaft sollen hier die Befehle MOVETO, DRAWTO und CLR (Bildschirm löschen) der Version G4 vorgestellt wer-

Zunächst muß die Tokentabelle erweitert werden. Problemlos geht dies bei der Tabelle TOK2B, also mit den Zwei-Byte-Token 80 XY. Die neuen Befehle werden einfach nach dem BYE-Token und vor dem Label FTXT47 eingefügt. Bild 1 zeigt die Ergänzung mit den neuen Token für CLR: 80 97, DRAWTO: 80 98 und MOVETO: 80 96. Es stünde auch das Token 80 90 zur Verfügung, da es nicht verwendet wird (FTXT47 = Einsprung für Fehlermeldung: nicht vorhanden).

```
dw    0           ;80 95  BYE

DW    MOVETO      ;80 96
DW    CLR         ;80 97
DW    DRAWTO      ;80 98

dw    FTXT47      ;  "nicht vorhanden"
```

Bild 1: Ergänzungen der Zwei-Byte-Token-Tabelle TOKB2

Damit diese Erweiterung berücksichtigt wird, muß die Abfrage in der Routine ab TOKAW1 der Befehl CP 16H geändert werden. Da drei neue Befehle hinzukommen, ergibt sich die neue Abfrage zu CP 19H.

Nun müssen die nach dem Befehl eingegebenen Parameter durch Unterroutinen übernommen werden. So sind bei MOVETO und DRAWTO die Koordinaten X und Y zu übernehmen. Das Monitorprogramm FLOMON des NDR-Rechners erwartet die X-Koordinate im HL-Register, die Y-Koordinate im DE-Register. Die Einsprünge sind für MOVETO die Adresse F046h und für DRAWTO F049h.

Die HEBAS-Routine AUSW07 holt eine nach dem BASIC-Befehl angegebene Koordinate ins DE-Register. Der Wert darf auch als Variable vorliegen. Da der Befehl z.B. MOVETO 100,120 lautet, muß ein Komma nach der ersten Koordinate folgen, sonst soll ein Syntaxfehler ausgegeben werden. Dies erledigt die Routine VGLAHL, im Akku ist das zu prüfende Zeichen. Über den Stapel werden die Koordinaten dann in die entsprechenden Register geladen und in die FLOMON-Routine gesprungen.

Bild 2 zeigt die Routinen der Befehle, die man der Übersicht halber am besten am Ende des Interpreters, allerdings noch vor dem Label BASPRO und der Byte-Definition DEFS 8 einschiebt, da sonst ein BASIC-Programm die neuen Befehle überschreiben würde. Beachten Sie bitte auch, daß vor dem Label BASPRO mindesten ein Byte den Wert 0 besitzt.

```
CLR:    CALL    CLRSCR ;Bildschirm lö-
                       schenRET

MOVETO:                ;MOVETO X,Y
                       ;X-Wert nach HL, Y-Wert
                       nach DE
        CALL    AUSW07 ;Zahl oder Variable holen
        PUSH    DE     ;X-Wert auf Stapel
        LD      A,2CH  ;folgt Komma
        CALL    VGLAHL ;sonst Fehler
        CALL    AUSW07 ;Y-Wert holen
        EX      (SP),HL ;HL retten u. Y-Wert nach
                       HL
        CALL    0F046H ;FLOMON MOVETO
        POP     HL     ;HL herstellen
        RET

DRAWTO:                ;DRAWTO X,Y
                       ;X-Wert nach HL, Y-Wert
                       nach DE

        CALL    AUSW07 ;Zahl oder Variabl
                       holen
        PUSH    DE     ;X-Wert auf Stapel
        LD      A,2CH  ;folgt Komma
        CALL    VGLAHL ;sonst Fehler
        CALL    AUSW07 ;Y-Wert holen
        EX      (SP),HL ;HL retten u. Y-Wert nach
                       HL
        CALL    0F049H ;FLOMON DRAWTO
        POP     HL     ;HL herstellen
        RET
```

Bild 2: Routinen für die Befehle CLR, MOVETO und DRAWTO

Zuletzt müssen noch in der Befehlstabelle TOKTB4 an der entsprechenden Stelle der Befehl und das Token eingetragen werden. Beim Befehl entfällt der erste Buch

stabe. Bild 3 zeigt die Ergänzungen. Nun kann man sich auch Befehle wie PAGE X,Y oder LINE X,Y,Z,U machen oder beliebig andere.

```
CCCC:                  ;Befehle mit C
db      0C3H
db      'LR'           ;CLR = Bildschirm
        löschen
db      80H,97H
db      'LEAR'
db      97H

DDDD:                  ;Befehle mit D
db      0C4H
db      'RAWTO'        ;DRAWTO X,Y
db      80H,98H
db      'ATA'
db      83H

MMMM:                  ;Befehle mit M
db      0CDH
db      'OVETO'        ;MOVETO X,Y
db      80H,96H
db      'AT'
```

Bild 3: Ergänzungen der Befehlstabelle TOKTB4

Quelle und Assembler?

Für den BASIC-Interpreter HEBAS steht der Quellcode auf Diskette zur Verfügung. Dieser besitzt einen Umfang von ca. 240 KByte und kann mit Standard-Textverarbeitungsprogrammen bearbeitet werden. Allerdings muß entsprechend Platz auf der Diskette sein, sonst dauern Such- und Austauschvorgänge sehr lange. Eine große RAM-Floppy (ab 512 K) ist zu empfehlen.

Das mit der Quelle ausgelieferte Ringbuch enthält das ausgedruckte PRN-File (über 530 KByte groß), also die Quelle des Interpreters mit den zusätzlichen Adressen bei den Labels, mit dem Kommentar und die Symbol-Tabelle. Auf der dazugehörigen Diskette sind die kommentierte Quelle und Symbol-Tabelle enthalten.

Als Assembler können der bekannte Z80-Assembler M80 von Microsoft oder der schnellere SLR180 vom Elektronikladen in Detmold Verwendung finden. Der SLR180 erzeugt wahlweise direkt ein COM-File ohne zusätzlichen LINK-Vorgang in 21 Sekunden (Quelle 235 KByte, 1 MByte RAM-Floppy), der M80 benötigt mehrere Minuten, zusätzlich muß noch der Linkvorgang durchgeführt werden.

Assemblieren...

Am einfachsten kopiert man den Quellcode (z.B. HEBAS.MAC), den Assembler und den Linker auf ein Diskette. Dann startet man den Assembler. Nach dessen Meldung mit einem Sternchen dürfen nur noch Großbuchstaben eingegeben werden, z.B. "HEBAS = HEBAS". Nachdem hoffentlich die Meldung No Fatal error(s) erschien, beenden wir mit CTRL-C und finden auf der Diskette die Datei HEBAS.REL vor.

...und linken

Nun wird der Linker aufgerufen. Auch er meldet sich mit einem Sternchen. Es erfolgt die Eingabe von "HEBAS,HEBAS/N/E" und es wird HEBAS.COM erzeugt.

Beim SLR180, den ich nur noch verwende, ist alles einfacher, man muß nur "SLR180 HEBAS" eingeben und erhält sofort HEBAS.COM. Wahlweise kann aber auch kompatibel zum M80 gearbeitet werden.

Interpreter-Baukasten:

In letzter Zeit erleben Z80-Steuerungscomputer einen ungeahnten Aufschwung. Hierfür gibt es keine höhere Programmiersprache, deren Quelle zur Verfügung steht und die in Größe, Befehlsumfang, Lage des Programmes, der Variablen u.a. beliebig veränder- bzw. anpaßbar ist und sowohl im EPROM als auch mit Diskettenbetriebssystem lauffähig ist. Zudem sollte die Entwicklung der Steuerungssoftware auf dem PC stattfinden.

Übrigens sind aufgrund eines Fehlers die vom Z80-MBASIC-Compiler speziell für ein EPROM erzeugten Basic-Programme in einem Steuerungsrechner nicht lauffähig (ein BDOS-Aufruf CALL 5 kann im EPROM nicht funktionieren).

Wunsch-Interpreter:
HEBAS ist beim Autor als Interpreter-Baukasten erhältlich. Zu einem Kernteil gibt es über 300 verschiedene Module als INCLUDE-Files, die die Basic-Befehlsroutinen enthalten. Bild 4 zeigt ein solches Modul.

```
;*************************************
;*** INCLUDE BUGRO.MAC
;*************************************
;
;Stand: 14.05.89

;BUGRO : aufgerufen von     : BSAW15
;BUGRO1: "          "       : BSAW24
;BUGRO2: "          "       : BSAW20
;BUGRO3: "          "       :OPTION,
                             FCB17, USER

BUGRO:                       ;ASCII-Test/
                             Umwandlung
        LD A,(HL)            ;Byte holen
BUGRO1: CALL BUGRO3          ;und testen
        JR BUZA0             ;Test Buchsta-
                             be oder Zahl
BUGRO2:                      ;Kleinbuchsta-
                             be x -> X

        LD A,(HL)  ;Byte holen
BUGRO3: CP 60H     ;wenn < 60H, dann
                   Großbuchstabe
        RET C      ;oder Steuerzei -
                   chen u. zurück
        CP 7BH     ;wenn >= 7BH,
        RET NC     ;dann auch zurück
        AND 5FH    ;sonst umwandeln
                   in Groß-
        RET        ;buchstabe u. zu-
                   rück
```

Bild 4: Eine Include-Datei der HEBAS-Modulversion

Der Kern des Interpreters bleibt bei allen Versionen gleich, er enthält die Tabellen und Sprungverteiler für die Befehle sowie die sog. BASIS-Files. Dies sind die "lebensnotwendigen" Routinen für den Interpreter.

Der Anfang des Interpreters beginnt bei der EPROM-Version mit der Sprungleiste für Ein-/Ausgabe-Routinen (entspricht der BIOS-Sprungleiste), hier muß der Anwender sein eigenes System einbinden, z.B. auch die Adressen der Portbausteine für die Abfrage von Drucktasten oder Ventilen.

Bild 5 zeigt, wie man eine serielle Schnittstelle für Terminalbetrieb einbindet.

```
z.B. für Baustein 8251

V24D:   EQU XY          ;DATENPORT
                        FUER 8251
V24S:   EQU ZV          ;STEUERPORT

INIT:   LD A,78         ;UART INITIALI-
                        SIEREN
        OUT (V24S),A    ;STEUERPORT
        LD A,5          ;FREIGEBEN
        OUT (V24),A     ;PORT 8251

V24OUT: PUSH AF         ;RETTE AKKU
LD A,C  ;ZEICHEN IN C
        OUT (V24D),A    ;AUSGEBEN
LOOP:   IN A,(V24S)     ;STATUS HO-
LEN
        BIT 0,A         ;TEST
        JR Z,LOOP
        POP AF
        RET

V24IN:  IN A,(V24S)     ;STATUS
        AND 2
        CP 2
        JP NZ,V24IN
        IN A,(V24D)     ;ZEICHEN HO-
LEN
        RET

CI:     JP V24IN
CO:     JP V24OUT
CSTS:   LD A,0          ;kein Status
        RET
```

Bild 5: Serielle Schnittstelle für Terminal-Betrieb

Alles möglich:
Je nach dem gewünschten Verwendungszweck kann mit diesem Baukasten ein Basic-Interpreter als automatisches System ohne Zugang durch den Anwender oder mit voll interaktivem Bildschirm-System mit Tastatur erstellt werden. Es können Programme an verschiedenen Speicheradressen im Eprom liegen, die beim Ablauf natürlich für Variablen etc. einen kleinen RAM-Speicher (minimal 2 KByte) benötigen.
Beim Autor gibt es eine Komplett-Version mit allen Quellen und Modulteilen für HEBAS (auch EPROM-Version) mit 2 Handbüchern, insgesamt fast 1,9 MByte an Software (auf Wunsch mit Z80-Emulator für PC) für 300.00 DM. Aber auch Teile davon sind erhältlich. Fast jedes Diskettenformat ist lieferbar!
Fordern Sie bitte die kostenlose, 27-seitige Produktinformation an (Dr. Hehl Hans, Lindenstr. 20, 8059 Wartenberg, 08762/3070).

HEBAS und messen:
In der Computerzeitschrift ELEKTOR, Heft 6/89, S.18, wurde ein Einplatinen-Meßcomputer mit der Z80-CPU veröffentlicht, der im Terminalbetrieb mit einer speziellen EPROM-Interpreterversion von HEBAS betrieben wird. Bekannte DFÜ-Programme wie PROCOM aus dem SHAREWARE-Bereich können als Terminalemulation auf dem PC verwendet werden und dienen zur Übertragung der Basic-Programme zu HEBAS. Heft 11/89 (Elektor) enthält Grundlagen zum Basic-Befehlssatz dieser Interpreterversion von HEBAS.

HEBAS und MSDOS:
Auch auf MS-DOS-Rechnern mit einem CP/M-Z80-Emulator kann der Interpreter HEBAS betrieben werden. Ab Sommer 1990 steht dann HEBAS auch mit 8086-Code zur Verfügung und ist IBM-kompatibel (ohne Grafik). Der Kernteil des Interpreters ist schon umgeschrieben und läuft auf jedem PC.

HEBAS und GW-BASIC
Die im ASCII-Format abgespeicherten Basic-Programme können von beiden BASIC-Interpretern geladen werden. Anschließend müssen HEBAS-spezifische Befehle wie z.B. die Grafik-Befehle in GW-BASIC-Befehle umgewandelt werden. Bild 6 enthält einen Vergleich einiger wichtiger Befehle.
Die verschiedenen Diskettenformate von MS-DOS und CP/M (NDR-Format) lassen sich mit im Handel erhältlichen Programmen konvertieren. Zu empfehlen ist z.B. SCOPY von ComFood, Am Rohrbusch 79, 4400 Münster-Roxel, Tel. 02534/7093, das ich selber verwende. So können am PC die MS-DOS-Befehle wie COPY etc. für CP/M-Disketten verwendet werden.

HEBAS:	GWBASIC:
CLR	CLS
PRINT#2	LPRINT oder OPEN "LPT1:" FOR OUTPUT
OPEN#10,"I","B:DATEINAME"	OPEN "I",#10,"B:DATEINAME"
ON EOF GOTO 200	IF EOF THEN 200
INPUT LINE #10,A$(I)	LINE INPUT #10,A$(I)
D$ = BYTE$(#0)	D$ = INKEY$
DIR	FILES
ERASE"B:DATEINAME"	KILL"B:DATENAME.BAS"
LOADGO "B:DATENAME"	LOAD"B:DATEINAME",R
KILL A$	ERASE A$
SWAP	EXCHANGE
RINGBELL	BEEP
OPTION#0,"W",40	WIDTH 40
LINE$ A,X,Y,X1,Y1	LINE (X,Y) - (X1,Y1)
BYE	SYSTEM

Bild 6: Vergleichsliste HEBAS und GWBASIC

Horst Joachim Fischer, Ulrich Kracker

Drehzahlmessung mit dem Z80-Aufbaupaket

Erinnern Sie sich an einen Artilel in der LOOP Nr. 12? Damals wurde eine Kamerea-Verschlußzeitmessung mit einem NDR-Minimalsystem vorgestellt. Das Besondere daran war, daß der Autor, Herr Fischer, eine Kombination von BASIC-Programm und Maschinen-Code einsetzte.
Die Vorteile, die daraus erwachsen, liegen eindeutig auf der Hand: Zeitkritsche Teile, wie das Auszählen von Impulsen, erledigt der Z80-Prozessor am genauesten, wenn er in seiner ureigenen Sprache gesagt bekommt, was er zu tun hat. Das Umrechnen und Darstellen dieser Zählergebnisse kann dann bequem in einer Hochsprache programmiert werden.

Dieses Konzept wurde bei dieser Applikation: Drehzahlmessung mit einem Magnetfeldsensor erneut erfolgreich eingesetzt.
Diesmal wurde das Programm wurde von Herrn Harry Wagner an der Berufsbildenden Schule in Meile geschrieben. Es wird im Unterricht in den Elektro- und FOS-Klassen vom Autor H. J. Fischer und von Herrr Wagner eingesetzt.

Hardware

Der Versuchsaufbau kann wie folgt vorgenommen werden:
Ein kleiner Permanent-Magnet wird an einer Motorwelle stabil befestigt. Ein Magnetfeldsensor wird so angebracht,daß er vom Magnetfeld des Permanent-Magneten bei einer Wellenumdrehung einmal geschnitten wird.

Der hier eingesetzte Magnetfeldsensor liefert dann direkt einen TTL-Impuls an der IOE-Baugruppe ab, das heißt, man benötigt außer der Betriebsspannung keine weiteren Schaltungsaufbauten. Nähreres über Magnetfeldsensoren erfahren sie an anderer Stelle in dieser LOOP.

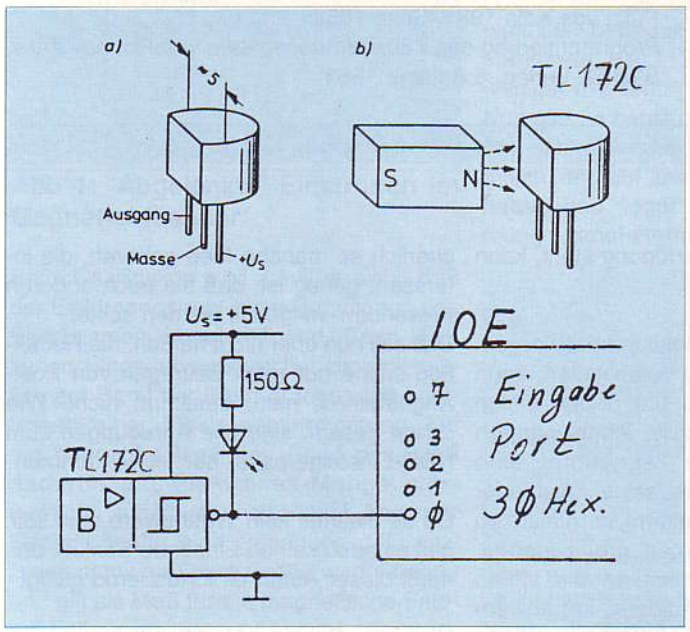


Abb. 1: Anschluß des Magnetfeldsensor TL172 C an die IOE

Assembler

Wird das BASIC-Programm nun gestartet, so wird zuerst das Assembler-Programm, das in dezimaler Darstellung in den DATA-Zeilen 70 bis 90 steht, in den Speicher geladen. Das Assemblerprogramm prüft nun dauernd den Wert des Bits 0 des Port 30H der IOE, an der der Magnetfeldsenor angeschlossen ist. Da das HL-Register das Zählregister ist, dessen Inhalt später die Drehzahl angibt, muß es als erstes mit der Anweisung auf 8F00H auf 0 gesetzt werden. Mit der UND-Verknüpfung in Adresse 8F05H werden die anderen Bits des Port 30H ausgeblendet. Bis Adresse 8F10H wird nun ein High/Low-Impuls vom Magnetfeldsensor "synchronisiert", damit man für die eigentliche Messung genau den Anfang der Periodenzeit erwischt.

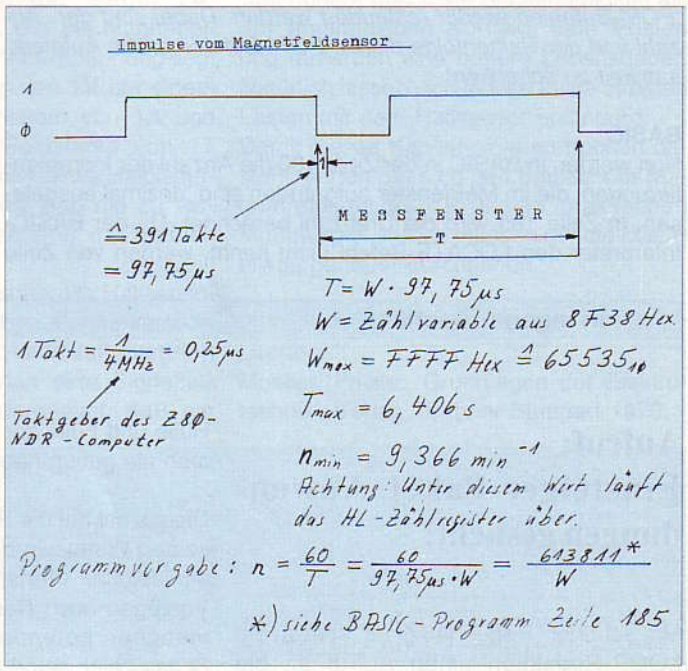


Abb. 2: Ausmessen eines Impulses über Software

Von Adresse 8F11H bis 8F2CH wird das Meßfenster geöffnet und das HL-Register alle 97.75 Mikrosekunden um 1 erhöht, bis der Magnetfeldsensor eine Amplitudenschwingung = eine Umdrehung der Welle abegtastet hat. Der Inhalt des HL-Registers wird nun auf Adresse 8F38H gerettet, bevor der nächste Meßzyklus beginnt.

```
A S S E M B L E R P R O G R A M M

Start-Adresse:8F00H

End-Adresse:8F30H
8F00 210000     LD HL,0000H
8F03 DB30       IN A,(30H)
8F05 E601       AND 01H
8F07 CA038F     JP Z,8F03H
8F0A DB30       IN A,(30H)
8F0C E601       AND 01H
8F0E C20A8F     JP NZ,8F0AH
8F11 23         INC HL
8F12 0E19       LD C,19H
8F14 0D         DEC C
8F15 C2148F     JP NZ,8F14H
8F18 DB30       IN A,(30H)
8F1A E601       AND 01H
8F1C CA118F     JP Z,8F11H
8F1F 23         INC HL
8F20 0E19       LD C,19H
8F22 0D         DEC C
8F23 C2228F     JP NZ,8F22H
8F26 DB30       IN A,(30H)
8F28 E601       AND 01H
8F2A C21F8F     JP NZ,8F1FH
8F2D 22388F     LD (8F38H),HL
8F30 C9         RET

Meßfenster

Die Anzahl der Takte für die
Befehle aus : "Programmierung
des Z 80" von Rodnay Zaks,
SYBEX-Verlag

6
7
4x25
10x25
11
7
10
391 Takte
```

Abb. 3: Das Assemblerprogramm zur Meßwerterfassung

Anmerkung der Redaktion:

Ein oftmals wichtiger Punkt ist bei diesem Assemblerprogramm nicht berücksichtigt worden, da es offensichtlich auf Anhieb problemlos funktioniert hatte: Das Retten der vom Programm veränderten Register-Inhalte. Da man im Allgemeinen nicht exakt weiß, welche Register vom übergeordneten (Hochsprache-) Programm belegt sind, sollte zu Beginn einer Assemblercode-Einfügung eine Reihe von 'PUSH'-Befehlen stehen. Am Ende des Programms müssen natürlich die entprechende Register mit 'POP'-Befehlen wieder restauriert werden. Dabei sind der Anzahl und der Reihenfolge dieser 'POP'-Befehle große Aufmerksamkeit zu schenken!

BASIC

Nun werden in BASIC in der Zeile 180 die Anzahl der Incrementierungen, die im Meßfenster aufgetreten sind, dezimal ausgelesen. In Zeile 185 wird die Drehzahl berechnet. Da der BASIC-Interpreter den LOCATE-Befehl nicht kennt, werden von Zeile 190 bis 220 die Tausender -, Hunderter-, Zehner- und Einerstelle ermittelt. Ab Zeile 230 bis 270 wird die Vordnullendarstellung ausgeblendet und von Zeile 290 bis 303 die Drehzahl auf dem Bildschirm dargestellt.

Dieser Vorgang wiederholt sich mit der aktuellen Drehzahl ständig, bis die Welle steht, oder das Programm abgebrochen wird.

```
5 REM ----------ELEKTRONISCHE DREHZAHLMESSUNG----------
7 REM----------Assemblerprogramm einlesen----------
10 FOR I=0 TO 48
30 READ CODE
50 POKE HEX("8F00")+I,CODE
60 NEXT I
70 DATA 33,00,00,219,48,230,01,202,03,143,219,48,230,01,194,10,143
80 DATA 35,14,25,13,194,20,143,219,48,230,01,202,17,143,35,14,25,13
90 DATA 194,34,143,219,48,230,01,194,31,143,34,56,143,201
100 REM ----------Deckblatt----------
110 CLRS
115 GOSUB 400
120 OUT 115,68
125 PRINT "E  l  e  k  t  r  o  n  i  s  c  h  e"
130 PRINT:PRINT:PRINT
135 PRINT "D  r  e  h  z  a  h  l  m  e  s  s  u  n  g"
140 GOSUB 400
145 OUT 115,17
150 PRINT"von OStR Harry Wagner, Berufsbildende Schule Melle"
160 FOR I=0 TO 10000
162 NEXT I
165 REM ----------Rechnen----------
170 CLRS
175 CALL HEX("8F00")
180 W=PEEK(HEX("8F38"))+PEEK(HEX("8F39"))*256
185 N=INT((613811/W)+0.5)
190 T=INT(N/1000)
200 H=INT((N-1000*T)/100)
210 Z=INT((N-1000*T-100*H)/10)
220 E=N-1000*T-100*H-10*Z
230 A$=STR$(T):B$=STR$(H):C$=STR$(Z):D$=STR$(E)
240 IF T=0 THEN A$=CHR$(32)
250 IF T=0 AND H=0 THEN B$=CHR$(32)
260 IF T=0 AND H=0 AND Z=0 THEN C$=CHR(32)
270 IF T=0 AND H=0 AND Z=0 AND E=0 THEN D$=CHR(32)
280 GOSUB 400
282 PRINT:PRINT:PRINT:PRINT
283 REM ----------Bildschirmausgabe----------
285 OUT 115,136
290 PRINT "D      r      e      h      z      a      h      l"
295 GOSUB 400
297 OUT 115,102
300 PRINT "n      =    ";A$;"    ";B$;"    ";C$;"    ";D$;
303 PRINT "      U    /    m    i    n"
310 OUT 115,17
320 FOR I=1 TO 1000
330 NEXT I
340 GOTO 165
350 END
400 FOR I=1 TO7
410 PRINT
420 NEXT I
430 RETURN
```

Abb. 4: Der BASIC-Überbau zur Drehzahlmessung

Literaturquellen:

- Einführung in die Digitaltechnik 2, herausgegeben von Jean Pütz, vgs Köln 1981, Seite 186ff
- Programmierung des Z80, herausgegeben von Rodnay Zaks, SYBEX-Verlag, 6.Auflage 1986

In eigener Sache

Aufruf: Einsteiger-Paket Anwendungen gesucht!

Mit schöner Regelmäßigkeit startet die LOOP-Redaktion einen Aufruf an die Anwender der Z80-Einsteigerpakete des NDR-Computers. Zuletzt wurde dieser Appell in der LOOP-Ausgabe Nr.15 im Rahmen eines Wettbewerbes verkündet. Die Reaktion darauf war sehr begeisternd und fruchtbringend.
Aber Anwendungsbeispiele für ein NDR-Z80-Minimalsystem für das Peripherie in Hülle und Fülle zur Verfügung steht, kann man nie genug haben!

Diesesmal hat die Redaktion beschlossen, keinen Wettbewerb zu veranstalten, denn ein solcher Wettstreit hat neben vielen Vorzügen auch Nachteile, indem nämlich mancher potentielle Teilnehmer eine Scheu zeigt, seine Idee, seine Arbeit oder Anwendung dem direkten Vergleich zu anderen Mitkonkurrenten preiszugeben. Wir denken vornehmlich an die vielen NDR-Einsteiger- und Aufbaupaket Anwender aus dem Kreise der Lehrgangsteilnehmer des M-I-K; das Fortbildungskonzept für Angestellte der Deutschen Bundespost. Im Rahmen dieses Lehrganges wird sicherlich so manche Idee geboren, die interssant genug ist, daß sie auch anderen Anwendern mitgeteilt werden sollte.
Das soll nun aber nicht heißen, die Redaktion suche nur nach Beiträgen von Post-Angestellten, nein, bestimmt nicht. Wie schon gesagt, sind alle Anregungen zum NDR-Einsteigerpaket höchst willkommen.

Da es diesmal kein Wettbewerb sein soll, gibt es auch keinen Einsende-Schluß, das heißt dieser Aufruf ist fortdauernd gültig!

Übrigens: Bei einer Veröffentlichung eines Beitrages winkt dem Autor ein Honorar von DM 50.- pro Textseite...
Also auf, 'ran an den Speck!

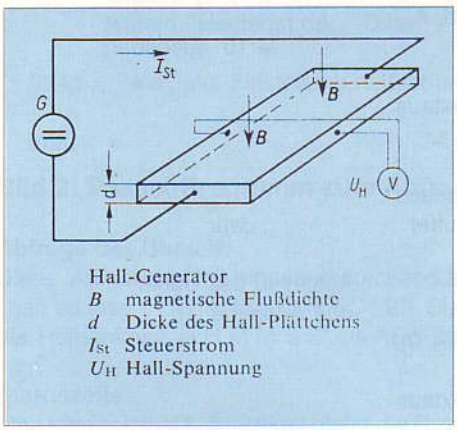
Ulrich Kracker

Magnetfeldsensoren

In der industriellen Meßtechnik stellt sich oft die Aufgabe, Magnetfelder zu messen. Warum, werden sich viele Leser fragen? Aufgrung der hohen Energiedichte, die mit Magnetfeldern erreicht werden können, werden diese für Leistungsantriebe bevorzugt eingesetzt. Schließlich arbeitet beinahe jeder Elektromotor und Lautsprecher auf der Basis des (Elektro-) Magnetismus, wogegen man zum Beispiel elektrostatische Felder hier nur sehr selten antrifft. Wenn man also in der Lage ist, das Magnetfeld in einer Maschine zu messen, kennt man auch die Leistungen, die hier umgesetzt werden.

Wirkungsweise eines Magnetfeldsensors

Ein Magnetfeldsensor beruht auf dem sog. HALL-Effekt und wird deshalb im übrigen auch HALL-Sensor genannt. Die grundsätzliche Arbeitsweise ist die gleiche wie bei der Elektronenstrahl-Ablenkvorrichtung in einer Braunschen Röhre (Fernseher, Oszilloskop). Hier wie dort werden freie Elektronen auf die Reise geschickt, was zunächst einem Stromfluß entspricht. Unterwegs geraten die Elektronen in ein quer zur Bewegungsrichtung angelegtes Magnetfeld und erfahren dadurch eine Krümmung ihres ansonten geradlinigen Bahnverlaufes. Im Physik-Unterricht der Gymnasialklassen ist es eine beliebte Aufgabe Krümmungsradien, Strom- und Magnetfeldstärken usw. zu berechnen.



Hall-Generator
B magnetische Flußdichte
d Dicke des Hall-Plättchens
I_{St} Steuerstrom
U_H Hall-Spannung

Abb 1: 'Abgelenkte' Elektronen im Magnetfeldsensor

Beim Oszilloskop wird damit erreicht, daß der Elektronenstrahl in die Randzonen des Bildschirmes umgelenkt wird. Beim Hallsensor hingegen entsprechen die sich seitlich am Rand der Bahn drängenden Elektronen einer Elektronenverdichtung. Da am gegenüberliegenden Rand der Bahn dadurch ein Elektronen-Mangel vorherrscht, erhält man also quer über der Stromflußbahn eine Spannungsdifferenz. Diese somit meßbare Größe wird versärkt und gilt als Maß für die magnetische Feldstärke.

Dieser Effekt ist grundsätzlich bei allen leitenden Medien feststellbar. Jedoch nur in Halbleitermaterialien ist er stark genug ausgeprägt, so daß sich eine technische Ausnutzung lohnt.

Aufbau eines Magnetfeldsensors

Die dem Elektronenstrom zur Verfügung gestellte 'Rennbahn' ist in der Regel weniger als 0.1mm dick und erstreckt sich über eine Fläche von wenigen Quadratmillimetern. Sie besteht aus Halbleitermaterial Idium-Arsenid oder Indium-Antimonid. Damit kann man also sehr kleine Sensoren aufbauen. Der Elektronenstrom in der Hauptflußrichtung wird mittels einer Hilfsspannungsquelle in der Regel zw. (3...20)mA eingestellt. Die bei Magnetfeldeinfluß abgreifbare HALL-Spannung liegt in der Größenordnung von 1V bei einem (hypotetischem) Bahnstrom von 1A und einer magnetischen Feldstärke von 1T (Tesla). Bezogen auf ein kleines Halbleiterplättchen bedeutet dies, daß die zu verstärkende HALL-Spannung im Bereich weniger mV liegt.

Nun gibt es heutzutage bereits Hallsensoren, die die notwendige Verstärkerstufe gleich mit auf dem Halbleiter integriert haben und sogar schon eine Signalformung vornehemen, dergestalt, daß am Ausgang direkt ein vom Computer verdaubares TTL-Signal abgenommen werden kann.

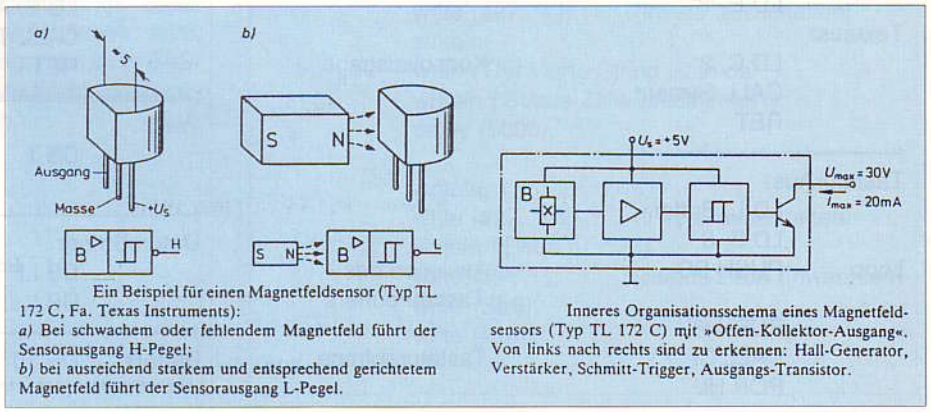
Im Vergleich zu anderen magnetempfindlichen Bauteilen (Induktionsspulen) hat ein Magnetfeldsensor (Hallsensor) den wesentlichen Unterschied, daß er nicht nur auf eine Änderung des Magnetfeldes, bzw. auf einer Änderung der vom Magnetfeld durchsetzten Fläche reagiert. Diese Eigenschaft wird überwiegend bei Messungen von langsam veränderbaren bis stillstehenden Magnetfeldern ausgenutzt (Dauermagnet, Erdmagnetfeld...). Eine Induktionsspule als Sensor kann hier nur schwer mithalten.

Viele Modelleisenbahner unter den Lesern werden sich jetzt vielleicht an die mechanischen REED-Relais errinnern. Richtig - aber: Gegenüber einem REED-Relais hat eine HALL-Sensor den wichtigsten Vorteil der kontaktlosigkeit. Das bedeutet eine weit höhere maximale Schalthäufigkeit als der mechanische Kontakt, kein Prellen und außerden eine höhere Lebensdauer. Natürlich lassen sich jedoch keine direkten Lasten mit dem Hallsensor ansteuern.

Damit dieses Kapitel nicht so trocken dasteht, sehen Sie sich den Einsatz der Magnetfeldsensoren in einem praktischen Meßbeispiel an! Sie müssen nur die LOOP etwas genauer durchsehen..

Literatur:

Moeller, Fricke: 'Grundlagen der Elektrotechnik', Verlag Teubner Stuttgart 1976.



Ein Beispiel für einen Magnetfeldsensor (Typ TL 172 C, Fa. Texas Instruments):
a) Bei schwachem oder fehlendem Magnetfeld führt der Sensorausgang H-Pegel;
b) bei ausreichend starkem und entsprechend gerichtetem Magnetfeld führt der Sensorausgang L-Pegel.

Inneres Organisationsschema eines Magnetfeldsensors (Typ TL 172 C) mit »Offen-Kollektor-Ausgang«. Von links nach rechts sind zu erkennen: Hall-Generator, Verstärker, Schmitt-Trigger, Ausgangs-Transistor.

Abb. 2: Ein HALL-Sensor mit TTL-Ausgang

Jost-Reimer Hoof

Einsatz von FLOMONCG-Monitor

Der neue Monitor FLOMONCG ist ein so vielseitiges Werkzeug, daß man nur nach einem intensiven Studium alle Möglichkeiten heraus finden kann. Wer hat aber so viel Zeit ? Deshalb soll in einer losen Fortsetzung über verschieden Einsatzmöglichkeiten von FLOMONCG berichtet werden. Jeder Leser ist aufgerufen, Beiträge einzusenden, von denen er glaubt, daß andere NDR-Computerbesitzer sie nutzen können.

Abfragen des Datums.
Mit der Eingabe "ESC ' 8'" läßt sich das Datum abfragen, das in Tastaturpuffer als ASC-Zeichen im Format "J J M M D D" bereit gestellt wird. Aber wie kann man diese Information heraus und in sein eigenes Programm einbinden? Bild 1 zeigt ein Beispiel in Assembler. Dabei sind zwei Unterprogramme wichtig:

```
                TITLE Datumsangabe      11.11.1989
;=============================================================
; Mit dieser Routine wird das Datum aus
; FLOMONCG ausgelesen und angezeigt.
;-------------------------------------------------------------
; Jost-Reimer Hoof    * LastUpdate : 13.11.1989
;-------------------------------------------------------------
System  equ     0005h
CI      equ     0F003h  ; Tastaturabfrage
LF      equ     10      ; LineFeed

        ORG 100h
        LD DE, DatumString      ; Datum-String ausgeben
        LD C, 9
        CALL System
        CALL DatumAbfrage       ; mit ESC ' 8'
        CALL TastaturAus        ; Tastaturpuffer in
                                ; Puffer auslesen
        CALL DatumAusgabe       ; Datum ausgeben
        RET
;=============================================================
DatumAbfrage:
        LD E, 1Bh               ; ESC ' 8' an System
        CALL Textaus
        LD E,' '
        CALL Textaus
        LD E, '8'
Textaus:
        LD C, 2                 ; Konsolausgabe
        CALL System
        RET
;-------------------------------------------------------------
TastaturAus:
        LD L, Buffer
        LD B, 6
Loop:   PUSH BC                 ; Auslesen des
                                ; Tastaturpuffers
        PUSH HL
        CALL CI                 ; Tastaturabfrage
        POP HL
        LD (HL), A
        INC HL
        POP BC
        DJNZ Loop
        RET
;-------------------------------------------------------------
DatumAusgabe:                   ; Ausgabe des Buffers
                                ; im Format: y y m m d d
        LD A, '.'               ; Wert ==> '.'
        LD (Wert), A
        LD IX, Buffer + 4       ; Tag
        CALL ASC_Aus
        LD IX, Buffer + 2       ; Monat
        CALL ASC_Aus
        LD A, ' '               ; Wert ==> ' '
        LD (Wert), A
        LD A, '1'               ; '19' ausgeben
        LD E, A
        CALL Textaus
        LD A, '9'
        LD E, A
        CALL Textaus
        LD IX, Buffer           ; Jahr

ASC_Aus:                        ; Auslesen des Buffer
                                ; und Ausgabe
        LD A, (IX + 0)
        LD E, A
        CALL Textaus
        LD A, (IX + 1)
        LD E, A
        CALL Textaus
        LD A, (Wert)
        LD E, A
        CALL Textaus
        RET
;=============================================================
Wert:
        DS 1                    ; benutztes
                                ; Zeichen (. : )
;-------------------------------------------------------------
DatumString:
        DB LF
        DB '    Heute ist : $'
;-------------------------------------------------------------
Buffer:  DS 10
;************************************ Ende ****
```

Bild 1: Abfragen des Datums in Assembler

Das Unterprogramm "Datumsabfrage" schickt "ESC ' 8' an den Rechner, während "TastaturAus" die Datums-Information in einen Speicher einliest, damit diese z.B. im Unterprogramm "DatumAusgabe" weiterverarbeitet werden kann. Wichtig ist, daß die Datums-Information sofort und in einem Zug in einen Hilfs puffer eingelesen wird, bevor sie weiter verarbeitet wird.
Bild 2 zeigt eine Prozedure in Turbo PASCAL, die die gleiche Aufgabe löst. Diese Routine wurde mir von Rüdiger Nahm zur Verfügung gestellt.

```
PROGRAM Datum;
        {*****************************************************
* Datum wird aus FLOMONCG ausgelesen und ausgegeben
*
* Ruediger Nahm (JRH)  *  LastUpdate 13.11.1989  *
*****************************************************}
{Variablendeklarierung}
VAR  clk  : ARRAY [1..6] OF Char;
     i    : integer;
     Date : STRING [8];

{ $u-,c-,  Turbo darf keine Zeichen verschlucken }

 Begin
        write(#27, ' 8');        { Datum auslesen }

        FOR i:=1 TO 6 DO Read (KBD, clk [i]);
        Date := clk [5] + clk [6] + '.' + clk [3]
              + clk [4] + '.' + clk [1] + clk [2];
        writeln;
        writeln ('Heute ist der ', Date);

{ $u+,c+,  evtl. alte Einstellung restaurieren }

 End.
```

Bild 2: Pasacl-Programm zum Abfragen des Datums

Abfrage der Uhrzeit
Diese Aufgabe ist mit einer entsprechend abgewandelten Routinen zu lösen. Der Aufruf ist "ESC ' 2'". Die Information steht in der Reihenfolge 'H H M M S S' als ASC-Zeichen bereit.

Statuszeile
Eine interessante Möglichkeit bietet die Nutzung der Statuszeile. Beispiele könnten sein, sich über den Fortschritt eines Programmes zu informiert oder sich zusätzliche Information z.B. über die Belegung von Sondertasten zu verschaffen. Bild 3 zeigt ein Beispiel, wie durch ein Programm an einer bestimmten Stelle das Wort "Ereignis 1" in die Statuszeile geschrieben wird, etwas später das Wort "Ereignis 2". Das Programm ist in PASCAL geschrieben und läßt sich als Procedure leicht umschreiben und in ein anderes Programm einbinden.

```
PROGRAM StatusZeileTest;
{*****************************************
* Testet die Status-Zeile des FLOMONCG    *
* Jost-Reimer Hoof * LastUpdate 11.11.1989  *
*****************************************}
CONST       Str1 = ' Ereignis 1  ';
            Str2 = ' Ereignis 2  ';
            Str3 = ' Ereignis 3  ';
            Str4 = ' Ereignis 4  ';
            Str9 = '                    ';

 Begin
            writeln ('Statuszeilen - Testprogramm ');
            w  r  i  t  e  l  n
('==============================JRH');
            writeln;
            write ('Mit diesem Programm wird die Statuszeile je nach');
            writeln ('nach Fortschritt');
            writeln ('des Programms mit einem String beschrieben.');

            write (#27, ')');       { Invers schalten }
            write (#27, 'f');       {Status-Zeile anwaehlen}
            write (Str1);
            write (#13);
            write (#27, '(');       { Invers ausschalten}
            writeln;
            write ('Jetzt ist der String in der Statuszeile');
            writeln ('erschienen');
            delay (3000);

            write (#27, ')');       { Invers schalten }
            write (#27, 'f');       {Status-Zeile anwaehlen}
            write (Str1, Str2);
            write (#13);
            write (#27, '(');       { Invers ausschalten }
            writeln;
            write ('Der zweite String ist in der Status-Zeile');
            writeln (' erschienen ');

            delay (3000);
            write (#27, 'j');       { Invers schalten }
            write (#27, 'f');       {Status-Zeile anwaehlen}
            write (Str1, Str2, Str3);
            write (#13);
            write (#27, 'k');       { Invers ausschalten }
            writeln;
            write ('Der dritte String ist in der ');
            writeln ('Status-Zeile erschienen ');
            delay (3000);

            write (#27, 'j');       { Invers schalten }
            write (#27, 'f');       {Status-Zeile anwaehlen}
            write (Str1, Str2, Str3, Str4);
            write (#13);
            write (#27, 'k');       {Invers ausschalten}
            writeln;
            write ('Der vierte String ist in der ');
            writeln ('Status-Zeile erschienen ');
            delay (5000);

            writeln;
            write (#27, 'k');       {Invers ausschalten}
            writeln ('Ende');
            write(#27, 'f');        {Status-Zeile anwaehlen}
            write (Str9);
            write (#13);
 End.
```

Bild 3: Procedure zum Ausgeben von Statusmeldungen

Bild 4 zeigt ein Programm in PASCAL, mit dem Informationen über Funktionstasten ausgegeben werden kann.

```
PROGRAM StatusZeileTest2;
{*********************************************
* Testet die Status-Zeile des FLOMONCG      *
* Jost-Reimer Hoof * LastUpdate 11.11.1989  *
*********************************************}
CONST  Str1 = ' ESC E ';
       Str2 = ' ESC A ';
       Str3 = ' ESC S ';
       Str4 = ' ESC B ';
       Str5 = ' ESC K ';
       Str9 = '|';
       Str8 = '                    ';
Begin
       writeln ('Statuszeilen - Testprogramm Nr. 2 ');
       w      r      i      t      e      l      n
('=================================JRH');
       writeln;
       write ('Mit diesem Progra#mm wird in die ');
       writeln (Statuszeile die Belegung ');
       writeln ('von Funktionstasten geschrieben.');
       write (#27, ')');          { Invers schalten }
       write (#27, 'f');          { Status-Zeile an-
       waehlen }

       write (Str1, Str9);
       write (Str2, Str9);
       write (Str3, Str9);
       write (Str4, Str9);
       write (Str5);
       write (#13);
       write (#27, '(');          { Invers ausschalten }
       writeln;
       delay (10000);

       writeln;
       write (#27, 'k');          { Invers ausschalten }
       writeln ('Ende');
       write(#27, 'f');           { Status-Zeile anwaehlen }
       write (Str8);
rite (#13);
End.
```

Bild 4: Pascalroutine zur Belegung von Funktionstasten

Walter Wild

MÄUSEKLAVIERE IN DER DUNKELHEIT

Nachdem mein NDR-Computer immermehr dazu benützt wurde, um die unterschiedlichsten Ausdrucke zu erstellen, ergab sich ein wachsendes Problem:
Mit dem jeweiligen Umbau von Endlospapier-Betrieb zum Einzelblatt-Betrieb und umgekehrt, wurde oftmals eine Veränderung der DIL-Schalter im Drucker erforderlich. Der Schalter leidet darunter und ist sehr oft nur schwer und in Dunkelheit erreichbar. Besonders, wenn man es eilig hat vergißt man einen Teil der Schalter, oder denkt erst daran, nachdem das Endlos-Papier eingelegt ist. - Mir scheint, Mäuseklaviere scheuen das Licht.
Deshalb habe ich mich kurz zum Schreiben dieser kleinen Programme entschlossen.
Das Umschalten des Druckers geschieht durch Aufruf des Programms für Endlos-Papier oder Einzelblatt-Magazin. Das Programm ist relokativ und unterstützt die CPU 68008 bis 68030.
Es kann Speicherresident geladen werden und wird über die Bibliothek gestartet - oder direkt von der Diskette mit dem Namen.
Da die Bildschirmtexte und auch die Steuersequenzen jeweils in Tabellen angelegt sind, ist eine beliebige Veränderung und Anpassung auch zu anderen Zwecken möglich.
Wer hier ändern möchte, ändert also nur die Tabellen und den Namen im Kopf, aber nicht das Programm. Die Länge der Tabellen ist beliebig und wird automatisch erkannt.
- Übrigens, hiermit ist es auch möglich dem Drucker die "Kursiv-Flausen" der Buchstaben ä,ö und ü auszutreiben. (Im Programm die Sequenz "MSB löschen").

```
* Druckertreiber um Drucker auf Einzelblatt umzuschalten.

KOPF:
 dc.l    $55AA0180
 dc.b    'DR-EINZ2'
 dc.l    DR-KOPF
 dc.l    DREND-KOPF
 dc.b    1
 dc.b    0,0,0
 dc.l    0,0


DR:
 movem.l d0-d7/a0-a1,-(a7) * Registerinhalte retten.
 clr.l   d0
 clr.l   d1
 clr.l   d2

 move  #!CLPG,d7        * Bildschirm löschen.
 TRAP  #1

 lea     TEXT1(PC),a0   * Adresse Text 1 nach A0.
 moveq #$22,d0          * Buchstabengröße definieren.
 moveq #0,d1            * Textlage definieren (linksbündig).
 moveq #127,d2          * Textlage in der Höhe definieren.
 move  #!WRITE,d7       * Textausgabe.
 TRAP  #1

 lea     TEXT2(PC),a0   * Adresse Text 2 nach A0.
 moveq #$21,d0          * Buchstabengröße definieren.
 moveq #100,d2          * Textlage in der Höhe definieren.
 move  #!WRITE,d7       * Textausgabe.
 TRAP  #1
```

```
 lea     TAB(PC),a0                      * Tabellenanfang laden.
 lea     TABEND(PC),a1                   * Tabellenende laden.

LOOP:
 move.b                  (a0)+,d0        * Wert aus Tabelle zum Drucker.
 bsr.s   LO
 cmpa.l a0,a1                            * Ende bei Tabellenende.
 bne     LOOP

 lea     TEXT3(PC),a0                    * Adresse Text 3 nach A0.
 moveq #$21,d0                           * Buchstabengröße definieren.
 moveq #20,d2                            * Textlage in der Höhe definieren.
 move  #!WRITE,d7                        * Textausgabe.
 TRAP  #1

 bsr     ZEIT                            * Zeitschleife für Anzeigetext (ist
                                         * bei Start über Romstart nötig).

 move  #!CLRSCREEN,d7                    * Bildschirm löschen für besseren
 TRAP  #1                                * Rücksprung zum JADOS.

 movem.l                 (a7)+,d0-d7/a0-a1  * Rückholen der Registerwerte.
rts


LO:
 move  #!LO,d7                           * Ausgabe zum Drucker.
 TRAP  #1
rts

ZEIT:
 move.l #$0003FFFF,d0                    * Zeitdauer in D0 festlegen.

LOOPZ:
 subi.l  #1,d0
 cmpi.l  #0,d0
 bne.s   LOOPZ
rts

*-------------------------------

TEXT1:
 dc.b    'DR-EINZ. V2.0 CPU 680XX, 1989 Walter Wild',0

TEXT2:
 dc.b    'Werte für Einzelbl.-Betrieb werden gesendet',0

TEXT3:
 dc.b    'Übertrag. der Einstellwerte ist beendet !!!',0

TAB:
 dc.b    27,64                           * Drucker normieren.
 dc.b    27,108,15                       * Linker Rand 15.
 dc.b    27,82,2                         * Deutscher Zeichensatz.
 dc.b    27,61                           * MSB löschen.
 dc.b    27,107,1                        * Sans Serif.
 dc.b    27,77                           * Elite.
 dc.b    27,67,61                        * 61 Zeilen.
 dc.b    27,25,4                         * Einzelblatt ein.
TABEND:

DREND:
end

* Druckertreiber um Drucker auf Endlosblatt umzuschalten.

KOPF:
 dc.l    $55AA0180
 dc.b    'DR-ENDL2'
 dc.l    DR-KOPF
 dc.l    DREND-KOPF
 dc.b    1
 dc.b    0,0,0
 dc.l    0,0


DR:
 movem.l                 d0-d7/a0-a1,-(a7)  * Registerinhalte retten.
 clr.l   d0
 clr.l   d1
 clr.l   d2

 move  #!CLPG,d7                         * Bildschirm löschen.
 TRAP  #1
 lea     TEXT1(PC),a0                    * Adresse Text 1 nach A0.
 moveq #$22,d0                           * Buchstabengröße definieren.
 moveq #0,d1                             * Textlage definieren
                                         * (linksbündig).
 moveq #127,d2                           * Textlage in der Höhe
                                         *  definieren.
 move  #!WRITE,d7                        * Textausgabe.
 TRAP  #1

 lea     TEXT2(PC),a0                    * Adresse Text 2 nach A0.
 moveq #$21,d0                           * Buchstabengröße definieren.
 moveq #100,d2                           * Textlage in der Höhe
                                         * definieren.
 move  #!WRITE,d7                        * Textausgabe.
 TRAP  #1

 lea     TAB(PC),a0                      * Tabellenanfang laden.
 lea     TABEND(PC),a1                   * Tabellenende laden.

LOOP:
 move.b                  (a0)+,d0        * Wert aus Tabelle zum Drucker.
 bsr.s   LO
 cmpa.l a0,a1                            * Ende bei Tabellenende.
 bne     LOOP

 lea     TEXT3(PC),a0                    * Adresse Text 3 nach A0.
 moveq #$21,d0                           * Buchstabengröße definieren.
 moveq #20,d2                            * Textlage in der Höhe
                                         * definieren.
 move  #!WRITE,d7                        * Textausgabe.
 TRAP  #1

 bsr     ZEIT                            * Zeitschleife für Anzeigetext (ist
                                         * bei Start über Romstart nötig).

 move  #!CLRSCREEN,d7                    * Bildschirm löschen für besseren
 TRAP  #1                                * Rücksprung zum JADOS.

 movem.l                 (a7)+,d0-d7/a0-a1  * Rückholen der Registerwerte.
rts


LO:
 move  #!LO,d7                           * Ausgabe zum Drucker.
 TRAP  #1
rts

ZEIT:
 move.l #$0003FFFF,d0                    * Zeitdauer in D0 festlegen.

LOOPZ:
 subi.l  #1,d0
 cmpi.l  #0,d0
 bne.s   LOOPZ
rts

*-------------------------------

TEXT1:
 dc.b    'DR-ENDL. V2.0 CPU 680XX, 1989 Walter Wild',0

TEXT2:
 dc.b    'Werte für Endlospapier werden gesendet !',0

TEXT3:
 dc.b    'Übertrag. der Einstellwerte ist beendet !!!',0

TAB:
 dc.b    27,64                           * Drucker normieren.
 dc.b    27,108,15                       * Linker Rand 15.
 dc.b    27,82,2                         * Deutscher Zeichensatz.
 dc.b    27,61                           * MSB löschen (wegen Umlaute).
 dc.b    27,107,1                        * Sans Serif.
 dc.b    27,77                           * Elite.
 dc.b    27,78                           * Randsprung ein
 dc.b    27,67,65                        * 65 Zeilen.
 dc.b    27,25,0                         * Einzelblatt aus.
TABEND:

DREND:
end
```

Günter Renner

Ärger mit Ä

Patchwork - 6. Teil

Böse Zungen behaupten, daß es einen sehr einfachen Weg gäbe, fast jeden Computerbenutzer in den Wahnsinn zu treiben. Man bräuchte ihm nur einen anderen Drucker zu geben. Dabei gehört doch gerade das Bearbeiten von Texten zu einem Haupteinsatzgebiet von Computern. Und eigentlich sollte man meinen, daß es hier keine Probleme mehr geben sollte. Mitnichten!

Für Eingeweihte ist der ASCII-Code ein Begriff. Ausgeschrieben nennt er sich 'American Standard Code for Information Interchange'.
Grob gesagt handelt es sich um einen Schlüssel, eine Tabelle, die den Buchstaben des Alphabets, den Ziffern und einer ganzen Reihe von anderen Zeichen, die man schreibenderweise braucht, bestimmte Byteinhalte zuordnet. Doch bereits hier fängt der Ärger auch schon an: verschiedene Länder haben verschiedene Zeichensätze, je nach den besonderen Charakteren, die sich landesspezifisch nennen. Bei uns in Deutschland sind es die Umlaute und das scharfe S.

Weil es in der Computerwelt aber nichts gibt, wofür es nichts gibt, haben sich die Fachleute auch hier etwas einfallen lassen. Sie haben die ASCII-Tabelle entweder kurzerhand erweitert, oder aber sie verwenden weit verwegenere eigene, mehr oder weniger von ihr abweichende Zeichensätze. Beides kann einem begegnen, wenn man es mit Textfiles zu tun hat. Leider stellt sich dieses Problem auch, wenn man Texte zwischen den IBM-kompatiblen PCs und dem NDR-Klein-Computer transferieren will. Beide Systeme verwenden erweiterte Zeichensätze, um die Umlaute darzustellen. Nur sind sie leider nicht dieselben!

Würde man etwa den Editor des RDK-Grundprogramms auf einen Text loslassen, der auf einem PC erstellt wurde und Umlaute enthält, gäbe es eine böse Überraschung: er wäre übersät mit Blockmarken, an allzu vielen Stellen würde der Editor kurzerhand ein Zeilenende setzen und dabei auch nicht davor zurückschrecken, Textteile ganz einfach zu entfernen.
Ähnlich enttäuschend, wenn auch weniger gefährlich, ist das umgekehrte Vorgehen.

Um all dem abzuhelfen, muß unser Diskmonitor auch Texte so aufbereiten können, daß sie auf dem jeweils anderen System editierbar und weiterverwendbar sind.
Allerdings ist dies nur mit gewissen Einschränkungen möglich. Denn neben dem Standard-ASCII-Satz kennt der NDR-Klein-Editor lediglich die Umlaute und das scharfe S als Sonderzeichen. Bei den PCs wird meist der sogenannte IBM-Zeichensatz #2 verwendet, und der enthält gleich 127 davon. Zum Glück ist der Großteil davon uninteressant. Es ist aber klar, daß nur die Zeichen, die es auf beiden Systemen gibt, in den Texten ersetzt werden können. Und genau das tun die vorgestellten Unterprogramme 'transndr' und 'transibm'. Dazu gibt es zwei Tabellen, die die in Frage kommenden Bytewerte enthalten.

Was aber tun, wenn nun in einem Text ein Sonderzeichen auftritt, das nicht ersetzt werden kann?

Dies könnte bei 'IBM-Texten' vorkommen, die für den NDR-Klein-Computer aufbereitet werden sollen. Nun, die entsprechenden Zeichen werden schlicht durch ein Leerzeichen ersetzt. Dasselbe gilt für das Tabulatorzeichen (Wert $09).

Wer nun meint, jetzt sei alles bestens, irrt ganz gewaltig. Denn es sind nicht nur die Sonderzeichen, die Sand im Getriebe eines Editors sein können. Die auf PCs üblichen Textverarbeitungssysteme können durchaus ihrerseits Zeichen hinterlassen, die stören.

Aus diesem Grund besteht das Programm 'transndr' bei genauerem Hinsehen aus zwei Teilen, zwei Schleifen, die jede für sich nacheinander das Textfile bearbeiten. Die erste ersetzt alle Umlaute durch die korrespondierenden Bytewerte; nicht übersetzbare Zeichen sowie das Tabulatorzeichen werden durch ein Leerzeichen ersetzt.

Die zweite Schleife entfernt zwei weitere Bytewerte aus dem Text, und zwar ersatzlos! Es handelt sich um $1b, auch als ESCAPE bekannt, und $1c. Der Grund liegt darin, daß Texte, die z. B. mit der Textverarbeitung WORDSTAR erstellt wurden, vor einem Umlaut oder einem anderen Sonderzeichen das Byte $1b und hinter denselben das Byte $1c enthalten können. Auch diese Bytes kann der RDK-Editor nun einmal nicht leiden, so daß sie weichen müssen. Nebenbei bemerkt verursachen diese Zeichen auch Störungen der MS-DOS-Systemfunktionen PRINT und TYPE, mit denen man sich Texte ansehen oder ausdrucken kann. Und vorsintflutlich anmutende Editoren, wie etwa EDLIN, haben ihre einzige Daseinsberechtigung allein darin, daß sie solchen Unfug gar nicht erst versuchen.

Es ist durchaus denkbar, daß andere Textverarbeitungen andere Kuckuckseier hinterlassen. Man müßte sie alle sehr detail-

liert kennen, um für alle Fälle Vorsorge treffen zu können. Im Zweifelsfall muß man sich den Text eben einmal in Hex-Darstellung ansehen, um zu erkunden, was Störungen verursacht.

Der RDK-Editor ist demgegenüber geradezu vorbildlich sauber. Hat man die Umlaute abgehandelt, kann eigentlich nichts mehr schiefgehen.

Ein weiterer kritischer Punkt bei der Bearbeitung von Texten ist das Ende derselben. Meist arbeitet man mit Endlosschleifen, die bei Auftreten eines bestimmten Bytewertes (Terminator, Delimiter) wieder verlassen werden. Natürlich hat man Pech, wenn das betreffende Byte einmal aus irgendwelchen Gründen fehlt. Aus Sicher-

heitsgründen ist deshalb ein Zähler für 64 kB vorgesehen, obgleich der RDK-Editor auch beim Auftreten des in PCs üblichen Delimiters $1a prompt anhält.
Im Menue des Monitors treten diese Funktionen als Programm 'editier' und 'textibm' in Erscheinung.
Mit 'editier' kann ein Text editiert werden, der auf einem PC erstellt wurde. Mit 'textibm' kann man solche, die mit dem RDK-Editor erstellt wurden und auf PCs weiterverwendet werden sollen, geeignet vorbereiten. Und selbstverständlich kann man damit auch einmal einem PC-Besitzer einen Brief per Disk schreiben...

Zum Schluß eine Tabelle der ersetzten Zeichen:

Char.	IBM#2	NDR
Ä	$8e	$db
Ö	$99	$dc
Ü	$9a	$dd
ä	$84	$fb
ö	$94	$fc
ü	$81	$fd
ß	$e1	$fe
Ende	$1a	$00

Patchwork-Listing Teil 6

```
****Menuefunktionen

editier:
bsr transndr            *Umlaute und EOF
lea filebuf(pc),a0
move.l a0,d0
move #!putstx,d7        *Textstart setzen
bsr traprts
moveq #$11,d0           *80 Zeichen/Z.
move #!size,d7
bsr traprts
move #!edit,d7          *Editor aufrufen
bsr traprts
moveq #$21,d0
move #!size,d7          *wieder 40 Z.
bsr traprts
move #!clrscreen,d7     *ganz wichtig!
bsr traprts
bra menue

textibm:
bsr transibm
bra menue


****Textprogramme

transndr:
lea filebuf(pc),a4
move #$10000-1,d1       *Zaehler 64 kB

transn1:
cmp.b #9,(a4)           *ist VTAB
beq transn4             *ersetzen
tst.b (a4)              *bit7 = 1?
bpl transn6

transn2:
move.b (a4),d0
lea tabibm(pc),a2       *Umlaut?
move #tabibm-tabndr-1,d2

transn3:
cmp.b 0(a2,d2.w),d0     *suchen/tabibm
beq transn5
dbra d2,transn3

transn4:
move.b #' ',(a4)        *'Dubiosa...'
bra transn6

transn5:                *ersetzen/tabndr
move.b tabndr-tabibm(a2,d2.w),(a4)

transn6:
cmp.b #$1a,(a4)+        *Ende?
dbeq d1,transn1

lea filebuf(pc),a4      *nochmal
movea.l a4,a3           *von vorn
move #$10000-1,d1


transn7:
cmp.b #$1b,(a3)         *ist ESC
beq transn8             *weg damit
cmp.b #$1c,(a3)         *das auch!
bne transn9

transn8:
addq.l #1,a3            *uebergehen
bra transn7

transn9:
move.b (a3)+,(a4)+      *nachruecken
cmp.b #$1a,-1(a3)       *Ende suchen
dbeq d1,transn7
clr.b -(a4)             *NDR-Ende
rts

transibm:
lea filebuf(pc),a4
move #$10000-1,d1

transi1:
tst.b (a4)
bpl transi4

transi2:
move.b (a4),d0
lea tabndr(pc),a2       *Umlaut?
move #tabibm-tabndr-1,d2

transi3:
cmp.b 0(a2,d2.w),d0     *suchen/tabndr
beq transi3a            *wenns passt
dbra d2,transi3
bra transi4

transi3a:               *ersetzen
move.b tabibm-tabndr(a2,d2.w),(a4)

transi4:
tst.b (a4)+             *Ende?
dbeq d1,transi1
move.b #$1a,-(a4)       *IBM-Ende
rts

tabndr:
dc.b $db,$dc,$dd,$fb,$fc,$fd,$fe

tabibm:
dc.b $8e,$99,$9a,$84,$94,$81,$e1

$94,$81,$e1


4ve.b d4,(a2)           *Motor an
bclr #6,d4
move.b d4,(a2)
move #16,d0             *'nicht gefunden'
and #$7f,d3
cmp #41,d3              *max. Spur?
bgt flop9               *ist Fehler!
bsr flop0               *Spur einstellen
bsr flop5               *1. Versuch
beq flop9               *gefunden?

bsr rstore              *wenn nicht
clr.b flo1              *Spur 0
bsr flop0               *neu anfahren
bsr flop5               *2. Versuch

flop9:
movem.l (a7)+,d7/a0-a3
rts

flop0:                  *Spur anfahren
cmp.b flo1,d3           *noch dieselbe?
beq flop3
blt flop2

flop1:
move d3,d7
sub.b flo1,d7           *SOLL-IST
subq #1,d7

flop1a:
bsr stepin
dbra d7,flop1a          *hinein!
bra flop3

flop2:
clr d7
move.b flo1,d7
sub.b d3,d7             *IST-SOLL
subq #1,d7

flop2a:
bsr stepout
dbra d7,flop2a          *heraus!

flop3:
rts

flop5:                  *ausfuehren
move.b (a1),d0
```

Patchwork - Listing Teil 6

Ralph Dombrowski

Das Grundprogramm Vers 6.2 ist fertig !!!

Seit ungefähr einem Jahr gibt es das Grundprogramm in der Version 6.0 für den 68008 und den 68000 bzw. 6.1 für den 68020. Seither haben viele NDR-Benutzer bei mir angerufen bzw. mir geschrieben, um mir Tips oder Anregungen zum Weitermachen zu geben. Vor allem wurde der Ruf nach einem Handbuch immer lauter, denn das Handbuch der Version 4.3. ist schon lange nicht mehr auf dem neuesten Stand. Dehalb ist es oftmals nicht bekannt, welche neuen Fähigkeiten es im Grundprogramm jetzt gibt. Nachdem ich erst einmal die Version 6.1 für den 68020 angepaßt habe (siehe LOOP 21) habe ich die Neuigkeiten auch für den 68008 und den 68000 geschrieben.

Außerdem habe ich in der Zwischenzeit an weiteren Verbesserungen und an Anpassungen für die neuen Baugruppen gearbeitet. Das Ergebnis liegt jetzt in der Version 6.2 vor, die nicht nur die neuen Baugruppen wie SCSI-Harddisk und Promer2 unterstützt, sondern auch in vielen Bereichen erweitert wurde. Außerdem ist jetzt ein sehr umfangreiches Handbuch in Vorbereitung, das alle Funktionen und Unterschiede zu alten Versionen erklärt. Damit dürften jetzt wirklich kaum noch Wünsche offen bleiben, und auch die Dokumentation ist jetzt wieder auf dem neuesten Stand.

Hier einmal kurz die wichtigsten Daten des Grundprogramms 6.2:

Alle drei Versionen für 68020, 68000 und68008 sind auf dem gleichen Stand und bieten (außer FPU bei 68020) die gleichen Möglichkeiten.

Der Assembler verfügt über eine gute-MACRO-Fähigkeit, wodurch Programme übersichtlicher und einfacher geschrieben werden können. Durch MACROs können auch leicht neue Pseudobefehle erstellt werden. Dies ist nötig für den geplanten JADOS-Linker. Weiterhin gibt es die Befehle RS, RSSET und RSRESET, mit denen Variablen unabhängig vom PC-Stand definiert werden können, wodurch es sehr viel leichter ist PC-unabhängige (relokative) Programme zu schreiben.

Der Editor verfügt über neue Befehle, die das Erstellen von Programmen erleichtern (Automatisches Einrücken, Ausdrucken des Bildschirmes, ...). Außerdem können vom Editor aus durch Angabe einer neuen Textadresse von jedem Betriebsystem aus mehrere Texte gleichzeitig bearbeitet werden.

Der HARDSCROLL mit der GDPHS ist jetzt in jedem Betriebssystem einsetzbar, da auch der Cursor damit darstellbar ist und Zeilen während des HARDSCROLLS bearbeitet werden können. Dadurch werden Ausgaben mit JADOS oder CP/M68K sehr viel schneller.

Im Menü erreicht man mit ESC immer eine höherer Ebene, wodurch die verschiedenen Rücksprungbefehle nicht mehr verwechselt werden können. Durch das Unterprogramm GRUND, das jetzt im Handbuch genau beschrieben ist, kann aber auch sehr leicht eine eigene Oberfläche programmiert werden, wodurch jeder Anwender frei nach seinem Geschmack mit dem NKC arbeiten kann. Dadurch ist zum Beispiel eine Oberfläche programmierbar, wie sie JADOS bietet, wobei die einzelnen Menüpunkte durch Eingabe eines Befehls aufgerufen werden. Diese Oberfläche kann natürlich auch unter JADOS aufgebaut werden, in dem die einzelnen Befehle mit dem Befehle INST installiert werden.

Neue Unterprogramme unterstützen eine beliebige SCSI-Harddisk. Dies ist Voraussetzung für die Benutzung des JADOS mit Festplatte. Außerdem kann direkt von der Harddisk gebootet werden, was den Start von CP/M oder JADOS sehr beschleunigt.

Eproms können nun wahlweise mit der alten Promer-Karte oder mit dem Promer2 programmiert werden. Der Promer2 kann nahezu alle Eproms der Typen 2716, 2732, 2764, 27128, 27256, 27512 und 27010 programmieren, wodurch auch für die Zukunft gesorgt ist. Damit können auch endlich Eproms für die ROA256 (27256 / 27010) gebrannt werden.

Es sind neue Unterprogramme für verschiedene Zwecke vorhanden. (Grafik-Routinen, System-Informationen, Zugriff auf Harddisk)

Die Hardcopys, die im Grundprogramm bisher erzeugt wurden, waren besonders beim Ausdruck mit 24 Nadeln sehr schlecht zu gebrauchen. Deshalb gibt es jetzt die Möglichkeit, die Größe des Ausdrucks zu bestimmen. Dazu wird ein getrennter Vergrößerungsfaktor für X- und Y-Richtung angegeben (0 bis 7).

Die Druckersteuerung enthält ein Feld, in dem bis zu 19 Bytes eingegeben werden können. Dadurch können Befehle hinzugefügt werden, die z.B. eine neue Schriftart einstellen o.ä. Außerdem ist bei der Diskettenversion ein Programm vorhanden, das die Druckereinstellungen laden und speichern kann, was ein erneutes Einstellen der Druckerbefehle nach dem Einschalten des Computers überflüssig macht.

Bei der Diskettenversion werden INCLUDE-Dateien mitgeliefert, die für jedes Unterprogramm im Grundprogramm einen standartisierten Aufruf in Form eines Macros enthalten. Das Grundprogramm benötigt zwei Disketten.

Im Handbuch ist jedes über TRAP aufrufbare Unterprogramm detailliert und mit mehreren Beispielen beschrieben. Jede Beschreibung enthält einen Kopf, aus dem man schnell die Eingabe- und Ausgaberegister, sowie die zerstörten Register entnehmen kann. Weiterhin ist jeder Menüpunkt genau beschrieben. Außerdem sind noch Kapitel vorhanden, die auf einige Spezialitäten des Grundprogramms eingehen (z.B Beschreibung der Symboltabellenstruktur oder des Bildschirmspeichers). Eine Aufteilung in Kapitel und Unterpunkte erleichtert den Umgang mit dem Handbuch und ermöglicht schnelles Finden der gewünschten Informationen.

Mit dem JADOS 3.3 steht jetzt ein noch leistungsfähigeres Betriebssystem zur Verfügung. Es unterstützt auch eine SCSI-Festplatte.

Das Grundprogramm 6.2 ist aber Vorrausetzung für diese JADOS-Version. Zusammen bilden sie ein starkes Gespann für den NDR-Klein-Computer, da sie in einigen Punkten aufeinander abgestimmt sind (Verlassen des Editors ohne Abspeichern, Laden von Texten vom Editor aus, Aufruf des Grundprogramms von JADOS aus, ...). Die grafische Benutzeroberfläche GRAFIK-JADOS bietet zusätzlich noch eine MAUS-gesteuerte Bedienung aller JADOS- und Grundprogramm-Funktionen an.

Uwe Koch

Grafik-JADOS

Nachdem das Standard-Betriebssystem für den NDR-Klein-Computer, JADOS, nun schon seit einigen Jahren zur Verfügung steht, gibt es jetzt mit "Grafik-JADOS" auch eine grafische Oberfläche für dieses System. Grafik-JADOS (im folgenden auch mit GJ abgekürzt) ist ein Zusatz zu JADOS, der ein menügeführtes Bedienen ermöglicht.

Zusätzlich zu den Standard-Funktionen des JADOS sind einige erweiterte Funktionen implementiert, die das Arbeiten mit dem NKC weiter vereinfachen. Grafik-JADOS enthält in der vorliegenden Version noch nicht alle Funktionen des JADOS-Kommando-Interpreters. Es wird aber ständig aktualisiert und neuen Grundprogramm- und JADOS-Versionen angepaßt.
Abb. 1 zeigt, wie sich Grafik-JADOS nach dem Aufrufen melden kann: Dateien mit gleichen Namenserweiterungen werden in Ordnern zusammengefaßt dargestellt.

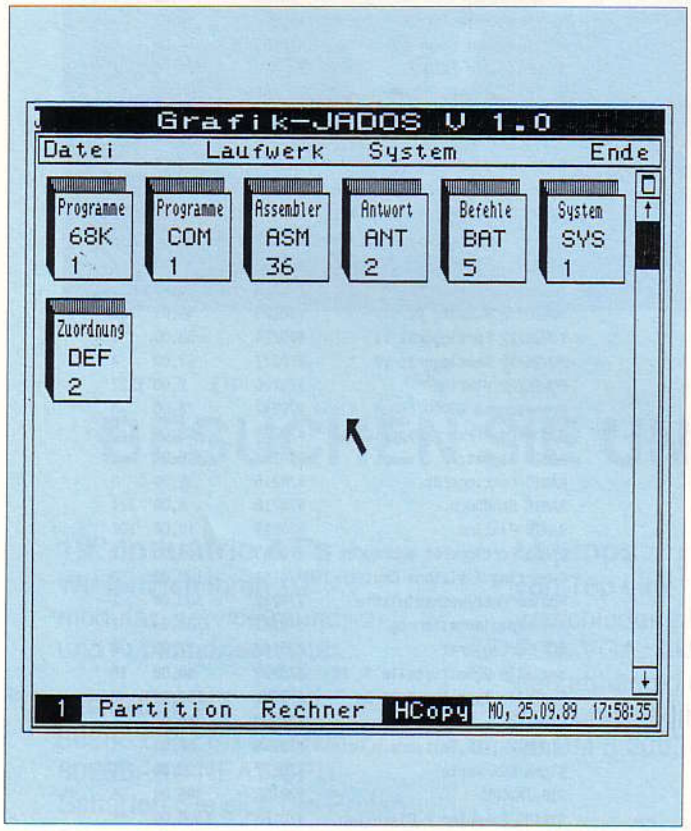


Abb. 1: D ie neue Grafik-Oberfläche unter JADOS

Folgende JADOS-Funktionen sind in GJ noch nicht implementiert :
- Aufruf von Batch-Programmen
- Installieren und Entfernen residenter Programme
- Assemblieren mit Antwortdatei
- Starten von COM-Programmen auf hohen Adressen

Grafik-JADOS bietet neben den Standard-JADOS-Funktionen, wie Starten, Anzeigen, Edieren, Löschen und Umbenennen von Dateien folgende zusätzliche Funktionen an :
- Menüführung
- Maus-Bedienung
- Festplatten-Unterstützung
- Erweiterte Druckfunktionen
- Drucker-Umleitung
- SER-Unterstützung
- Terminal-Betrieb
- Datei versenden über serielle Schnittstelle
- Datei empfangen über serielle Schnittstelle
- Disketten-Kopie
- Erweiterte Laufwerksinformationen
- Grundprogramm-Aufruf
- Umfangreiche Hardware-Information
- UHR-Unterstützung
- Mehrere Hardcopy-Möglichkeiten
- Integer-Rechner bei 68008 und 68000
- Fließkomma-Rechner bei 68020/68881
- Individuell einstellbare Konfiguration
- Ausführliche Fehlermeldungen

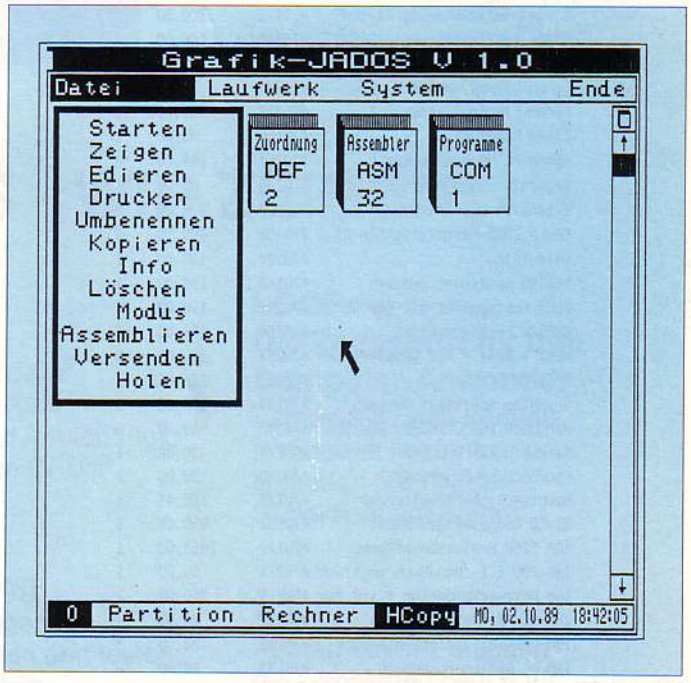


Abb. 2: Die wichtigsten Arbeiten mit Dateien können nun grafisch eingeleitet werden.

Für den Betrieb von Grafik-JADOS sind folgende Hard- und Software-Voraussetzungen notwendig :
- CPU 68008 oder CPU 68000 oder CPU 68020/68881
- GDP-64-HS (da RMW-Einsatz)
- mindestens 128 kByte RAM Arbeitsspeicher
- mindestens 1 Disketten-Laufwerk
- EGRUND 6.0 oder höher
- JADOS 3.01 oder höher

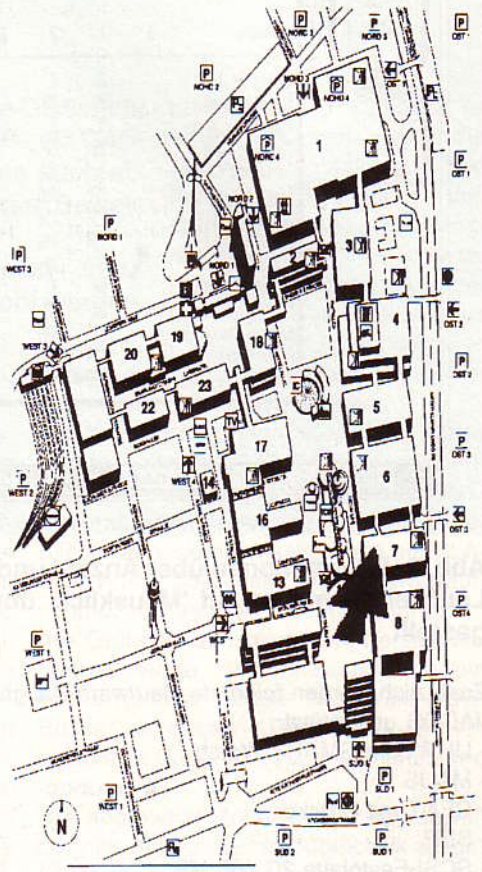
Aus unseren Sonderangeboten

Bezeichnung	Artikelnummer	Preis	Anzahl
24 poliger Sockel	#70209	0,82	9999
30MB Festplatte AT03	#70188	900,00	1
384KB Speichererweiterung o. R	#70285	336,30	1
5250 Emulationsprogramm deutsc	#70169	890,00	1
68000 Grundlagen Teil 1	#70244	50,00	15
68008 Aufbau Listing	#70234	10,00	145
68008 Grundprogramm Listing	#70233	10,00	115
Adapter f. Graphik/Farbbildsch	#70196	147,99	3
Adapter f. Monochrom Bildschir	#70195	97,99	1
BANKSEL Bausatz	#70214	30,00	7
BANKSEL Platine	#70215	10,00	196
Bopla-Gehäuse 718	#70149	35,00	68
C-Programmierung	#70241	49,00	7
C64ADAPT Bausatz	#70230	40,00	12
C64ADAPT Fertiggerät	#70229	60,00	3
C64ADAPT Handbuch	#70232	5,00	442
C64ADAPT Platine	#70231	25,00	220
CPU68K Fertiggerät	#70266	50,00	4
CRT1 Fertiggerät	#70220	150,00	10
Data-Becker,Inferf.f.EP-Dr.an	#70280	59,00	1
dBase II CP/m84 Stdt. IBM	#70200	364,00	1
Der sichere Einstieg in Pascal	#70240	35,00	3
Disketten 8", SD/SS Verbatim	#70204	7,90	110
DOS2.0 Handbuch deutsch	#70137	40,00	3
DOS2.1 Handbuch deutsch	#70138	40,00	8
Einführung in LOGO	#70239	40,00	31
Einstieg in C	#70238	65,00	3
Einzelblattzuführung FX-100	#70166	100,00	1
Einzelblatteinzug f. LX80/90	#70180	250,00	1
Einzelblatteinzug für NT Druck	#70192	698,00	1
Einzelblattzuführung RX-100	#70175	100,00	1
Einzelblattzuführung RX80F/T	#70176	100,00	2
Eprom-Floppy Bausatz	#70221	99,00	6
Eprom-Floppy Handbuch	#70223	10,00	364
Eprom-Floppy Platine - 2	#70222	20,00	42
Epson Kabel TRS80	#70159	59,00	1
Epson PC Monochrom Karte	#70267	149,00	2
Ergotilt Jumbo Monitorständer	#70235	85,08	1
Extension Unit extern	#70179	279,00	2
FD-LW 20MB Festplatte für XT-2	#70189	400,00	2
Filevision	#70194	100,00	1
FILING Assistant deutsch	#70140	298,00	1
FLO2 Fertiggerät mit KAB008	#70247	198,00	1
GDP64K Fertiggerät r1	#70254	198,00	19
GEH2 + BUS4 + NE2 komplett ver	#70257	390,01	8
GEH2+BUS3F	#70260	249,00	1
Graphing Assistant deutsch	#70144	199,00	3
Handbuch für CP/M68K, deutsch	#70243	49,00	8
Handbuch FX+ Drucker	#70277	30,00	1
Handbuch LX-80 Drucker	#70278	30,00	1
Hauptspreichererweiterung	#70173	394,44	1
HX-20 Computer gebraucht	#70152	950,00	1
IBM 5250 Emulationsadapter	#70170	2468,00	1
IBM DOS 1.1, Handbuch englisch	#70271	30,00	1
IBM Netzwerkprogramm V.1.0 deu	#70273	250,00	1
IBM PC AT Inst. Handbuch	#70132	20,00	4
IBM PC AT03 Bedienerhandbuch	#70136	20,00	3
IBM PC Bedienerhandbuch	#70133	20,00	6
IBM PC DOS 3.10 deutsch	#70272	159,00	1
IBM PC DOS 3.2 deutsch	#70258	150,00	1
IBM PC Finanzbuchhaltung Schul	#70145	698,00	4
IBM PC XT Bedienerhandbuch	#70135	20,00	1
IBM Portabel Tastatur	#70167	498,00	1
IBM Scanner 3117/011 m. Erw.Ei	#70284	4402,68	1
IBM-PC Emulation FX100+	#70147	49,00	4
IBM-PC Emulation LQ-1500	#70146	97,00	2
IEEE-Parallel I/F	#70160	129,00	2
Imagwriter Acc. Kit Mac	#70203	68,40	1
Interface Centr.Parallel 8171	#70163	199,00	3
Interface Epson 8120	#70161	78,00	1
Interface für AppleII Text	#70156	50,00	10
Interface Graphik für Apple II	#70158	50,00	5
Interface seriell für LQ1500	#70178	348,00	1
Kabel EPSON IEEE	#70157	47,00	2
Kabel für Apple Graphik 8132	#70153	73,01	14
Kabel für Applekarte 8131	#70154	43,00	10
Kabel IBM-Epson 90cm	#70208	10,80	11
KEY Fertiggerät r3	#70255	69,00	13
kompatible CGA Karte	#70274	100,00	1
Lohn und Gehalt Schule	#70150	500,00	2
Lotus 1-2-3 deutsch	#70134	500,00	1
M68000-Familie, Teil 2	#70245	50,00	14
MFII-TAST wenig Steuerz,fl.bel	#70282	50,00	1
MFII-Tastatur funktionsf.,kl.F	#70281	149,00	4
Microsoft DOS 3.2 deutsch	#70279	248,00	1
Microsoft-Chart 65 IBM PC DOS	#70131	350,00	1
Mono Graphic Printer Card	#70171	97,99	2
Monochrom Bildschirm	#70185	150,00	3
MS-Flight Simulator PC DOS IBM	#70207	50,00	1
Nachfüllband Epson FX-RX-Serie	#70165	18,00	8
Nachrüstsatz FX100+	#70155	104,00	1
NEC P7 Bedirektionaler Traktor	#70191	500,46	1
Open Access, IBM PC/XT	#70201	500,00	2
Option zu Megaplus	#70168	185,82	5
OS/2 Standard Edition d V.1.0	#70290	400,00	2
PC Programm Schreibmaschinenku	#70206	195,72	1
PC-HD Computer IBM	#70177	2500,00	2
Planen u. Kalkulieren m. Multi	#70237	49,00	4
Planning Assistant deutsch	#70142	199,00	2
POW26/22 Bausatz V1	#70225	39,00	8
POW26/22 Bausatz V2	#70228	55,01	65
POW26/22 Fertiggerät V1	#70224	68,00	9
POW26/22 Fertiggerät V2	#70227	97,00	4
POW26/22 Platine	#70226	(!) 2,00	1227
Prozessoren 68000/68008, Klein	#70242	78,00	6
PX8 Handy Text + Handy Calc	#70275	159,00	2
RAM16 Bausatz	#70217	20,00	1
RAM16 Fertiggerät	#70216	30,00	8
RAM16 Handbuch	#70218	5,00	324
RAM16 Platine	#70219	10,00	102
RAM64 Fertiggerät gebraucht	#70263	149,00	2
Reporting Assistant deutsch IBM	#70141	199,00	2
Rodime Fest/Wechselplatte	#70212	500,00	2
Rollenpapierhalterung	#70181	33,99	1
SER Fertiggerät	#70249	97,00	8
serielle Schnittstelle f. PC	#70293	40,00	18
Serielle Tastatur 2400 Bd	#70291	109,00	3
Sidekick+ 31/2"	#70287	259,92	1
SIDHA-WRITER,Textvera.m.Datenf	#70276	50,00	1
Sigma EGA Karte	#70172	200,00	1
SIP-IATCPU	#70283	395,00	5
SIRIUS Computer 2 Disketten	#70183	1500,00	1
Smarty Modem 1203	#70286	950,00	1
STRONG-Prigrammiersystem IBM	#70151	300,00	1
Syquest SQ 360R	#70199	298,00	4
TA DINA3 Schreibmaschine SE1	#70289	2450.00	1
TextStar 3.0	#70261	500,00	1
TURBO 80186	#70174	570,00	1
TURBO Toolbox dt 65 IBM PC DOS	#70182	120,00	1
UNICAD	#70162	350,00	1
VFEATURE s/W f. 32MB DOS-GR.	#70259	710,79	1
Wordlord Textverarbeitung	#70202	130,01	1
WORDSTAR 2000 deutsch	#70148	900,00	1
Writing Assistant deutsch	#70143	199,00	1
XENIX-Betriebssystem englisch	#70139	498,00	2

GRAF® computer

HANNOVER MESSE
CeBIT'90
Welt-Centrum Büro · Information · Telekommunikation
21. – 28. MÄRZ 1990

HALLE 7 - Stand A03



BESUCHEN SIE UNS - WIR ZEIGEN:

19"-Industrie AT's
Wir fertigen Ihren 19" AT modular, servicefreundlich und in Industriequalität...

modular AT
80386 CPU mit 25 MHz
80286 - 16 NEAT CPU
Schauen Sie sich die Preise an...

mic
modulare Industrie Computer
Das Ausbildungssystem für gehobene Ansprüche
Ideal für Industrie- und Schulausbildung
modular aufgebaut
Standard ECB - Bus

Laptops
von Top Link
verschiedene Modelle
EGA-VGA, 2MB RAM
netzabhängig und -unabhängig
mit verschiedenen CPU's
ab DM 8.200,--

Logsim/Profilog
Digitale Schaltungen am Bildschirm erstellen und testen, und danach mit dem...

PC-Interface - Multi I/O
Experimente aufbauen und somit den PC mit der Außenwelt verbinden.

LOOP -
Die Zeitschrift für den Computer-Anwender
Lernen Sie Loop kennen (Probeheft kostenlos) oder sprechen Sie mit der Redaktion: am Stand!

A.L.F.
Auftrag-Lager-Fertigung
die komplette Lösung für produzierende Unternehmen
Demo-Diskette kostenlos erhältlich

vereinbaren Sie einen Termin !

Achtung: Jede Stunde verlosen wir einen Logsim - schauen Sie vorbei !

Graf Elektronik Systeme Magnusstraße 13 8960 Kempten Tel.: (0831) 6211 Fax: 61086

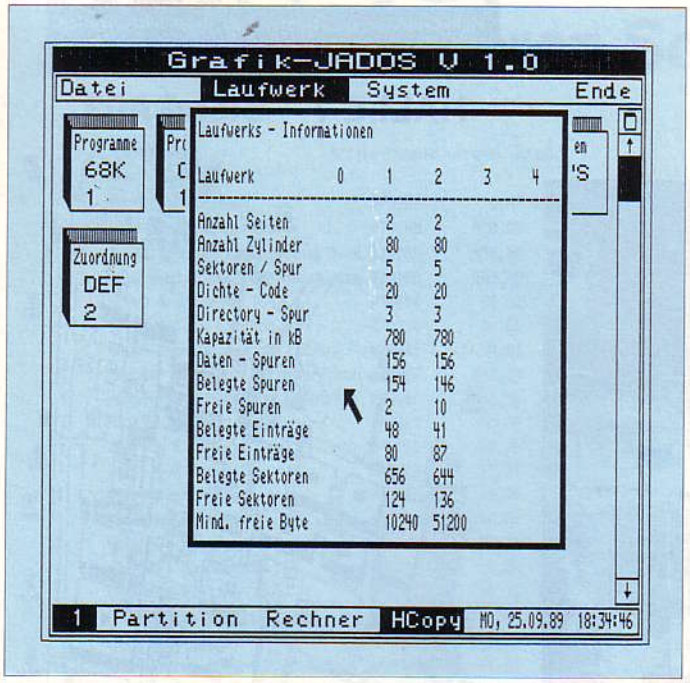


Abb. 3: Informationen über Anzahl und Aufteilung der Laufwerke werden auf 'Mausklick' übersichtlich dargestellt.

Zusätzlich werden folgende Hardware-Baugruppen von Grafik-JADOS unterstützt:
- UHR oder SMART-Watch
- MAUS
- CENT mit Drucker
- SER
- SCSI-Festplatte 20 - 65 MB
- 24 kByte RAM hinter Grundprogramm für Hardcopies

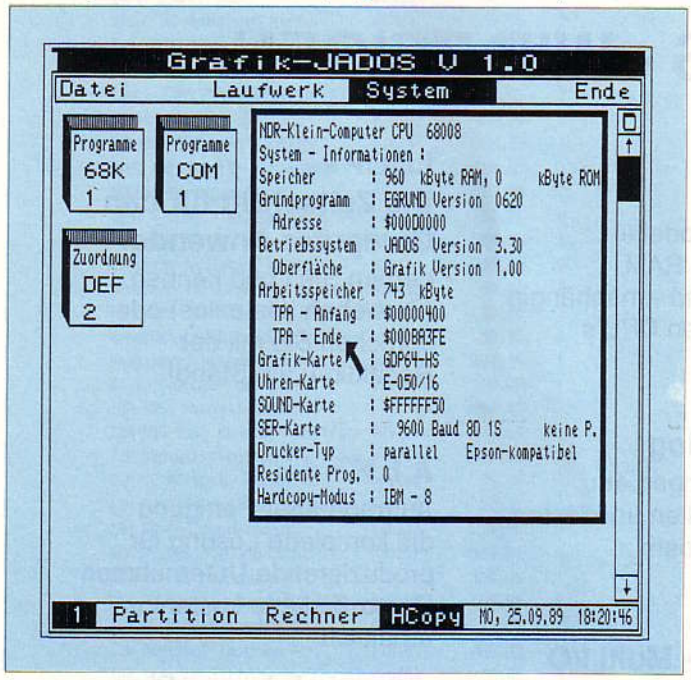


Abb. 4: Weitreichender Zugriff auf die übrige Systemkonfiguration, erstmals auch auf die SER-Baugruppe

Softwareseitig ist ein Einsatz von EGRUND 6.2 und JADOS 3.3 zu empfehlen, da nur hiermit alle Funktionen im vollen Leistungsumfang zur Verfügung stehen.

Alle Funktionen von GJ lassen sich durch Eingaben auf der Tastatur oder durch die Auswahl mit der Maus erreichen. Dateinamen können direkt aus dem Inhaltsverzeichnis übernommen werden, so daß bei Arbeiten mit bestehenden Dateien die Tastatur garnicht benötigt wird.

Abb. 5: System-Informationen auf einen Blick, das bietet Grafik-JADOS

Nach dem Start von GJ erscheint das Hauptmenü (siehe Abb. 1). Das Hauptmenü enthält die Untermenüs für die Funktionen "Datei", "Laufwerk", "System", "Ende", "Partition", "Rechner" und "Hardcopy".

Wie der Leser in Abb. 2 ersehen kann, enthält das Untermenü "Datei" alle Funktionen zum Bearbeiten einzelner Dateien, wie z.B. Anzeigen, Edieren, Drucken, Löschen usw. Unter "Laufwerk" sind die Funktionen erreichbar, die sich auf ganze Laufwerke beziehen, und unter "System" die Funktionen, die das Rechnersystem selbst betreffen (siehe Abb. 3 und Abb. 4).
Die Funktion "Partition" ist abhängig von der vorhandenen Version von JADOS und dient der Auswahl von Disketten oder Festplatten-Partitionen. Die Funktion "Rechner" enthält einen CPU-abhängigen Integer- oder Fließkomma- Rechner.

Last but not least können mit dem Menüpunkt "HCopy" Auszüge vom momentanen Arbeitsbildschirm angefertigt werden, was zu Zwecken der Dokumentation oftmals sehr hilfreich ist. Die Abbildungen 1...4 wurden zum Beispiel auf sehr einfache Weise mit diesem Menüpunkt angefertigt.

Anmerkung der Redaktion:
Bei entsprechendem Interesse der 68000er-Anwender ist GES gerne bereit dieses, wie wir meinen sehr gut ausgereifte Produkt, den Vertrieb aufzunehmen. Da Grafik-JADOS, wie der Name schon andeutet, eng mit dem Betriebssystem zusammenhängt, könnte beides zusammen als Paket (z. B. JADOS 3.5 und Grafik-JADOS) auch sehr sinnvoll sein. Wie ist Ihre Einstellung? Wäre das lukrativ für Sie? Wenn Sie uns Ihre Meinung mitteilen wollen, kurze Postkarte oder Telefonanruf (0831/6211 H.Kracker) genügt.

GRAF-Computer in der Forschung

Daß mc modular-AT Computer aus dem Hause GES nicht nur auf dem Schreibtisch ihre Funktion als DTP-, als CAD- oder als Verwaltungscomputer im Netz, sowie auch als 'Standalonesysteme' ihre Zuverlässigkeit unter Beweis stellen, ist beileibe nichts Neues mehr.
Auch den gehobeneren Qualitätsansprüchen der Industrie kann der mc modular AT durchaus gerecht werden.

Dies beginnt beim Einsatz des obligatorischen 19"-Gehäuses und endet nicht zuletzt bei der individuellen Kundenentwicklung, sowie der detaillierten Angabe von Systemspezifikationen. Der Umstand des modularen Aufbaus des Computers kann im industriellen Einsatz als wesentlicher Vorteil gegenüber herkömmlichen Motherboardsystemen genannt werden. Beim Ausfall einer Teilkomponente ist das Gesamtsystem durch den simplen Austausch der betreffenden Funktionseinheit umgehend wieder in Betrieb zu nehmen. Damit werden teure Ausfallzeiten auf ein absolutes Minimum reduziert.
Was der Industrie als wesentlich erscheint, ist in der Forschung gerade recht: Zuverlässigkeit und flexible Systemkonfiguration. Aus diesem Grund kam beim nachfolgend beschriebenen Forschungsprojekt ein Rechnerverbund aus dem Hause GES zum Einsatz.

Besseres Energie-Management mit Computerhilfe

Um den zunehmenden elektrischen Energiebedarf der kommenden Jahre auch zu Spitzenlastzeiten ohne den aus vielen Gründen umstrittenen Bau von Kraftwerken decken zu können, bietet sich eine bessere Steuerung des Energiebedarfs über die Zeit an. Diese Steuerung kann dabei in beliebigen Größenordnungen ansetzen. Für die Zukunft ist abzusehen, daß nicht nur in Industriebetrieben eine rechnergestützte Lastverteilung vorgenommen wird. Es ist denkbar, daß bis herab zu den Haushalten der Bedarf an elektrischer Energie von einem selbstlernenden System unter dem Gesichtspunkt einer gleichmässigen Auslastung des Stromversorgungsnetzes gesteuert wird. In Ansätzen wird dies heute schon mit Tonrundsteueranlagen verwirklicht.

Lernen am Modell

Um diese Zusammenhänge studieren zu können wurde ein Modellversuch geschaffen, anhand dessen ein selbstlernendes System entwickelt werden kann, das zunächst die Optimierung des Wirk- und Blindleistungsverbrauches innerhalb eines Zeitraumes übernimmt. Als einen ersten Schritt in diese Richtung soll mit der Modellanlage jedoch erst einmal ausschließlich Blindstromkompensation nach frei vorgebbaren Kriterien betrieben werden. Die dafür benötigte Modellanlage besteht demzufolge aus zwei Hälften: Dem energietechnischen Teil und dem Mikroelektronischen. Die Realisierung der Modellanlage wurde im Rahmen zweier Diplomarbeiten vorgenommen: Die Projektierung und Errichtung eines Stromrichters mit einer Nennleistung von P = 15kW und der Option des Rechneranschlusses wurde an der Hochschule der Bundeswehr in München [1] vorgenommen. Die darauf aufbauende Auslegung und der Aufbau eines Rechnersystemes, sowie die Koppelung wurde an der Fachhochschule in Kempten [2] durchgeführt. Zu diesem Zweck wurde die Modellanlage in das Labor der Firma GES transportiert. Da die Leser der LOOP wohl mehr an der mikroelektronischen Seite des Projektes interessiert sein dürften, wird nun kurz dieser Teil vorgestellt:

HOST- SLAVE System steuert Modellanlage

Bei der Auswahl der eingesetzten Rechner wurde von folgender Überlegung ausgegangen: Einerseits bedarf es einer Rechnereinheit, die schnell genug sein muß, den Prozess zu überwachen und andererseits sollte die Möglichkeit bestehen, verschiedene Regelagorithmen auszutesten und zu optimieren. Aus Gründen der einfachen Strukturierbarkeit sollten diese Regelalgorithmen in einer Hochsprache formuliert werden.
Als Konsequenz hieraus wurden diese beiden Funktionsblöcke zwei verschiedenden Rechnern zugewiesen: Ein HOST-Rechner übernimmt die Regelung der Blindleistungskompensation und ein SLAVE-System übernimmt die Meßwertaufnahme, sowie deren Vorverarbeitung.
Aus Gründen der universellen Konfigurierbarkeit wurde für das Subsystem ein 68020-Rechner auf der Basis des NDR-Busses eingesetzt. Das HOST-System besteht im Gegensatz dazu aus einem **mc modular-AT.**
Die komplette Modellanlage ist, soweit möglich, in 19-Zoll Einschubtechnik ausgeführt und in zwei entsprechenden Schränken untergebracht. Das bedeutet, daß auch das NDR-Subsystem in ein 19"-Einschubgehäuse eingesetzt werden mußte. Das NDR-Subsystem besteht aus folgenden Peripheriebaugruppen:

- CPU-Baugruppe mit 68020 und 68881, Bildschirmadapter und Tastaturan-

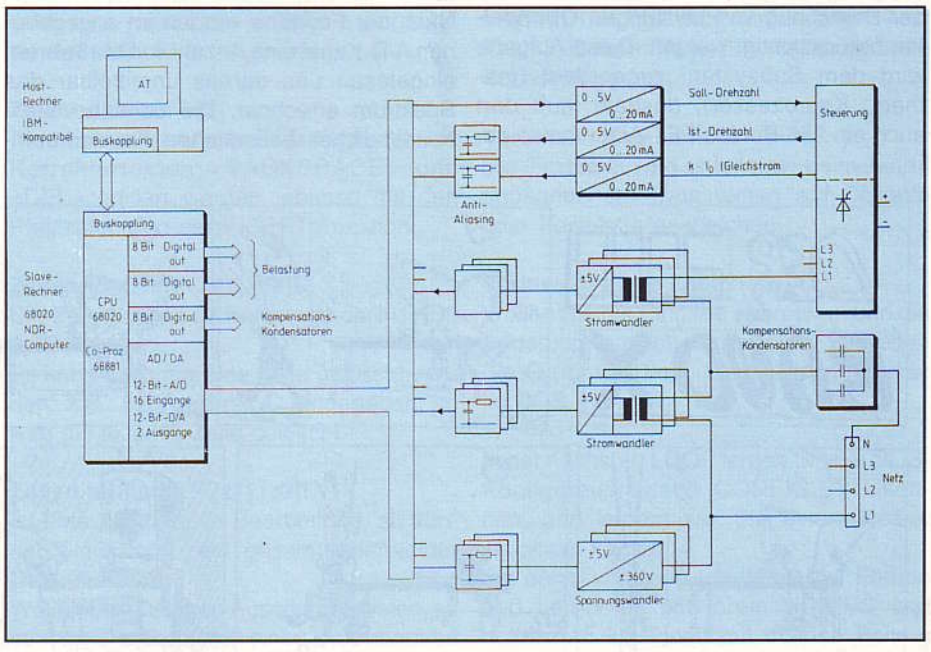


Abb. 1: Übersicht der Rechnerkonfigurationen

schluss (GDP und KEY) zur Erprobung der Subsystemsoftware, kann später entfernt werden,
- 3 * Relaiskarte (REL) je 8 Relais
- Analogwandlerbaugruppe (ADDA) 16*12Bit Eingabe, 2*12Bit Ausgabe,
- Buskoppelbaugruppe (NDRKOPP) für DMA-Kommunikation mit Hostrechner.

Der HOST-Rechner weist demgegenüber eine relativ einfache Ausstattung auf:
- 16 Bit 286er CPU mit einer Taktfrequenz von 10MZ,
- 20 MB-Festplatte, 1 Floppylaufwerk 1.2MB
- Grafik-Bildschirmadapter (EGA).
- Buskoppelbaugruppe (IBMKOPP) für DMA-Kommunikation mit Subsystem.

Software: Zusammenarbeit Host-Slave
Wie oben schon angedeutet wurde, teilen sich HOST- und SLAVE- Rechner die Arbeit:
Der Eine übernimmt die Koordination des Gesamtablaufes (wie komplex dieser auch sein mag), der Andere die Meßwerterfassung und die Meßwert-Vorverarbeitung. Diese Aufteilung läßt sich durch nachfolgendes Schema verdeutlichen (Abb. 2).

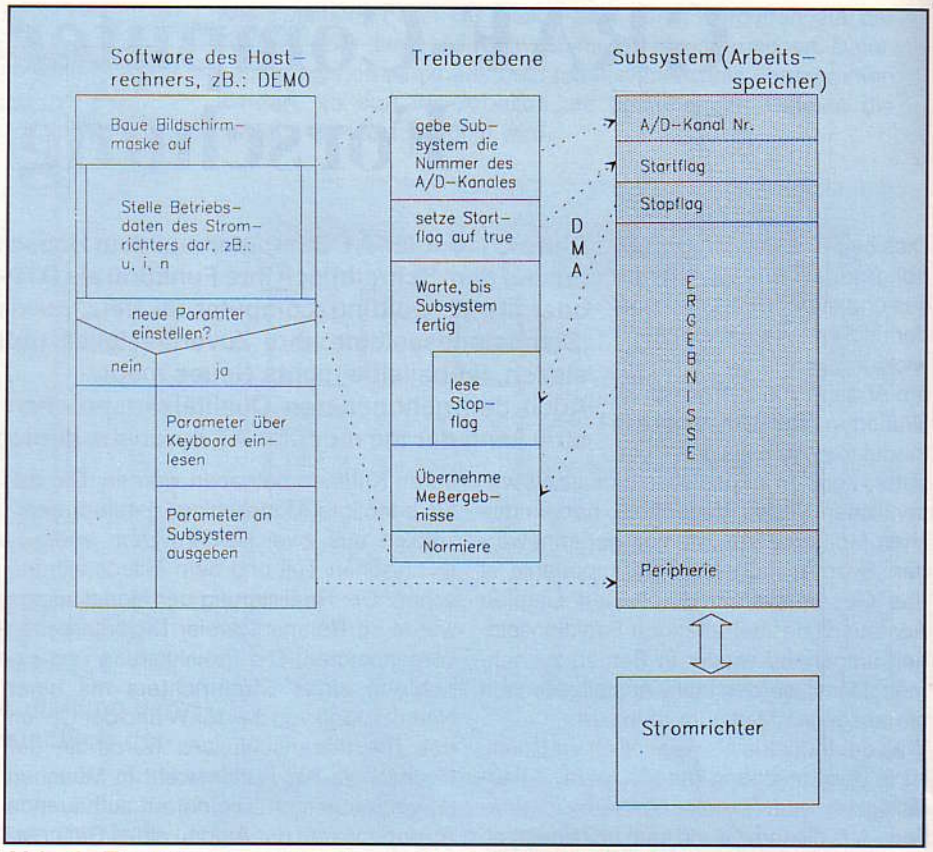
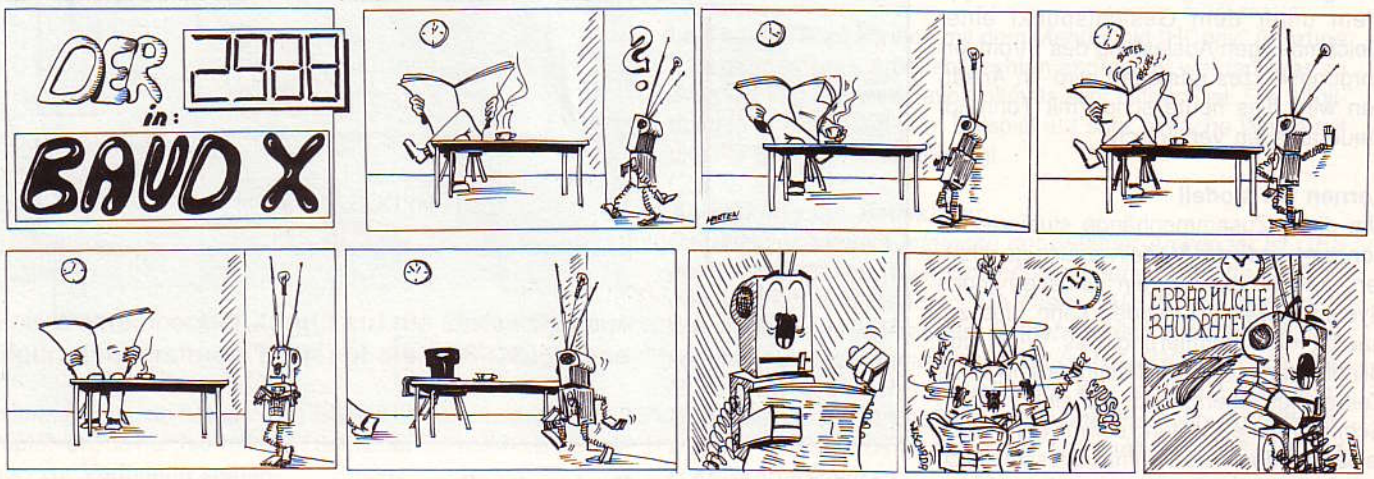


Abb. 2: Zusammenspiel von Haupt- und Nebenrechner

Die Programme des Hauptrechners (modular-AT) sind in Turbopascal 4.0 geschrieben, während die Routinen des Subsystemes (NDR) in Assembler-Sprache erstellt sind. In einer Vorstufe soll der HOST-Rechner lediglich die momentane Blindleistung ermitteln und anzeigen. Desweiteren sollen alle wichtigen Betriebsparameter der Modellanlage grafisch aufbereitet werden. Da im allgemeinen Fall im Energieversorgungsnetz **keine** ideale Sinusgrößen zu erwarten sind, müssen bei der Berechnug von Leistungen **Oberwellen** berücksichtigt werden. Diese Aufgabe wird dem Subsystem zugeordnet (mathem. Koprozessor), deshalb läuft dort auch ein 256-Punkte FFT-Algorithmus ab.

Eine vollständige Berechnug der 128 Fourierkoeffizienten nach Betrag und Phase dauert ca. 55ms. Über spezielle Flags im Speicherbereich des NDR-Rechners wird das erforderliche 'Softhandshake' abgewickelt. Über den DMA-Kanal (BUSKOPP) kann der Hauptrechner diese Speicherstellen lesen, bzw. manipulieren. Der NDR-Computer wartet zunächst einmal auf 'grünes Licht' vom Hauptrechner, bevor er mit der Abarbeitung einer Endlosschleife fortfährt.
Nach der Freigabe werden im angegebenen A/D-Kanal eine Anzahl von Meßwerten eingelesen und daraus unmittelbar das Spektrum errechnet. Die danach im Arbeitsspeicher befindlichen Fourierkoeffizienten können auf ein Zeichen des NDR-Computers hin vom **mc modular-AT** ausgelesen werden. Zu einem späteren Zeitpunkt kann sich der HOST-Rechner während dieser Zeit natürlich seinen 'höheren' Regelungs- und Optimierungsaufgaben zuwenden. Damit sollte einmal umrissen werden, wie sich der sinnvolle Einsatz einer DMA-Kopplung mit industrietauglichen Rechnern demonstrieren läßt.

Literatur:
[1] Bruno Tonelli: Demonstrationsanlage zur Blindleistungskompensation', Diplomarbeit an der Universität der Bundeswehr München 1988.
[2]Ulrich Kracker: Analoge und digitale Meßwerterfassung mit Rechnerkopplung und FFT', Diplomarbeit an der Fachhochschule Kempten 1990.



Volker Stahl

CPU8088 - MS/DOS

Teil 3: EDLIN.COM

Sicherlich haben Sie schon einaml im Inhaltsverzeichnis Ihrer DOS-Programmdiskette die COM-Datei 'EDLIN' gefunden und vielleicht sogar schon einmal gestartet. Dann erscheint die Meldung "Neue Datei *" und nach erfolglosem herumprobieren haben Sie verärgert einen Reset ausgelöst und fragten sich, was das überhaupt sein soll?!

EDLIN ist eine ausführbare Programmdatei auf der DOS-Diskette. Diese Datei ist ein Texteditor, vergleichbar mit dem RDK-Editor oder JEDI, aber, so finde ich, wesentlich unkomfortabler. Wer also ernsthaft Programme oder Texte (Briefe) schreiben möchte (oder muß), sollte sich ein proffesionelles Textverarbeitungsprogramm zulegen (z.B. Wordstar).
Wozu kann man dann EDLIN.COM benützen? - Nun ja, im Grunde genommen nur um kleinere Texte zu schreiben oder zum Ausprobieren. Ich verwende EDLIN ausschließlich um meine Batch-Dateien zu erstellen (wird in LOOP Teil 5: AUTOEXEC.BAT behandelt).
Doch wollen Sie dennoch den DOS-Editor verwenden, sollten Sie auch die Handhabung von EDLIN.COM sowie die wichtigesten Befehle kennen!
Um eine schon vorhandene Textdatei bearbeiten zu können, müssen Sie folgendes Kommando bei MS/DOS eingeben:

```
EDLIN dateinamen
```

Natürlich muß dazu der Editor EDLIN und die zu bearbeitende Textdatei auf der selben Diskette sein! - Ebenfalls ein großer Nachteil: EDLIN.COM ist nicht speicherresident oder ein interner Befehl (wie zum Beispiel der Befehl CLS). Deshalb greift der 'DOS-Commander' bei der Eingabe von EDLIN immer auf das angemeldete Arbeitslaufwerk und lädt EDLIN.COM (und gegebenfalls die zu bearbeitende Textdatei, bei unserem vorigen Beispiel die Textdatei dateiname). Nach der Eingabe erscheint die Edlin- Meldung 'Ende der Eingabedatei *'. Nun können Sie die Editierbefehle Eingeben und Ihren Text 'dateiname' verändern oder ergänzen!

Wollen Sie eine neue Textdatei anlegen, so geben Sie nur EDLIN ein, und der Editor meldet sich mit dem Hinweis 'Neue Datei *'. Jetzt können Sie wieder Editierbefehle eingeben.

Die wichtigsten Befehle werde ich jetzt im einzelnen aufführen, doch zunächst noch eine Zusammenfassung der Befehle und deren Syntax:

[z]I	Zeilen einfügen Ihre Zeile[z] editieren
[z1],[z2]D	Zeilen von z1 bis z2 löschen
[z1],[z2]L	Zeilen von z1 bis z2 auflisten
[z1],[z2]<z>C	Zeile z1 bis z2 nach z kopieren
[z1],[z2][?]S[text]	Text suchen
E	Editor verlassen, Änderungen sichern
Q	Editor verlassen, ohne speichern

Zeile einfügen - [n]I
a) bei neu angelegter Datei:
ist dies der erste Befehl, zur Einleitung des Textes.

b) bei bestehender Datei :
kann eine neue Zeile in Zeile z eingefügt werden; z.B. 3I

Zeile editieren - [z]
Die mit z aufgerufene Zeile kann mit den Korrekturtasten <BACKSPACE> und <DEL> editiert werden, ebenso mit den Pfeiltasten und der INSERT-Funktion.

Zeilen löschen - [z1],[z2]D
Die Zeilen z1 bis z2 werden gelöscht; z.B.: 3,7D.
Es kann auch nur eine Zeile gelöscht werden; z.B.: 5D (wird nur D eingegeben, so wird die aktuelle Zeile gelöscht)!

Zeilen auflisten - [z1],[z2]L
Ist eine Textdatei in Bearbeitung, so können Sie durch L den gesamten Inhalt der Datei auflisten.
Wollen Sie nur einen Auschnitt auflisten, so können Sie dies durch einen Zeilenangabe machen; z.B.: 6-20L

Zeilen kopieren - [z1],[z2],<z>C
Um Tipparbeiten ersparen zu können, können Sie einen Zeilenbereich z1 bis z2 nach (vor) Zeile <z> kopieren; z.B.: 1,3,8C

Text suchen -[z1],[z2][?]S[text]
Die Textsuche kann durch einen angegebenen Zeilenbereich z1 bis z2 begrenzt werden. Geben Sie diese nicht an, so wird die gesamte Textdatei nach text untersucht.

Wird noch zusätzlich ein ? eingegeben, so fragt EDLIN bei dem gefundenen Text 'O.K. ?' Geben Sie jetzt N ein, so wird die Suche fortgesetzt.

Verlassen und sichern - E
Die Textdatei wird unter dem beim Aufruf angegebenem Dateinamen auf Diskette oder Festplatte gespeichert.

Verlassen ohne speichern -Q
Wollen Sie die Eingabe abbrechen und die Änderungen nicht abspeichern, so geben SIe Q (für Quit) ein. Sie sind dann wieder im DOS.

In der nächsten LOOP lernen Sie die DOS-Konfigurationsdatei CONFIG.SYS kennen, und lernen, wie Sie einen Treiber installiern können.
Bis dahin wünsche ich Ihnen viel Freude und Lernerfolg mit Ihrem NDR-PC und hoffe, daß ich Ihnen mit meinen Beitrag helfen konnte.

Klaus Janßen

FESTPLATTENBETRIEB UNTER JADOS 3.5

Das bewährte 68000-Betriebssystem JADOS wurde kräftig überarbeitet und liegt nun als Version 3.5 vor. Die neue Version unterstützt den Betrieb einer SCSI-Festplatte zwischen 20 MByte und 65 MByte. Dazu kommen noch zahlreiche Detailverbesserungen wie Konfigurationsdatei, wahlfreier Dateizugriff, Kommandointerpreter als Unterprogramm usw.

Festplattenunterstützung

Bisher unterstützte JADOS den Betrieb von maximal vier Diskettenlaufwerken und einer Ramdisk. Der Betrieb mit Floppies ist von der Kapazität einigermaßen ausreichend aber recht langsam. Mit der Ramdisk kann man zwar sehr schnell arbeiten, doch leider kann sie nur im normalen Arbeitsspeicher eingerichtet werden; außerdem sind die Daten nach dem Abschalten des Rechners verloren, es sei denn, man hat eine Akkupufferung eingebaut. Der ideale Massenspeicher muß also schnell sein und eine hohe Kapazität anbieten; eine Festplatte muß her!

Die Hardwarelösung war schnell gefunden. Eine preiswerte Festpatte von SEAGATE mit eingebautem SCSI-Controller läßt sich mit einer leicht modifizierten SASI-Baugruppe leicht ansteuern. Nachdem dann noch Ralph Dombrowski sein Grundprogramm so erweitert hatte, daß er das BIOS zur Festplatte liefert und im Menu Folppystart auch die Festplatte erscheint, habe ich die Möglichkeit genutzt und JADOS so erweitert, daß es die Festplatte ordentlich verwaltet. Inzwischen möchte ich die Festplatte nicht mehr missen!

JADOS ist so aufgebaut, daß es einen Massenspeicher in Teilbereiche, logische Spuren genannt (in MSDOS heißt es Cluster), aufteilt. Diese Spuren werden in einer sogenannten Spurtabelle verwaltet. Während im Inhaltsverzeichnis nur die Startspur einer Datei verzeichnet ist, kann JADOS mit Hilfe der Spurtabelle die weiteren Spuren, die zu der Datei gehören, finden. Dabei sind die Spuren mit einer doppelt verketteten Liste verbunden. Das heißt zu jeder Spur gibt es einen Vorgänger und einen Nachfolger. Bei Disketten faßt JADOS je 5 Sektoren zu einer Spur zusammen. Bei einer 800Kbyte- Diskette ergeben sich also 160 Spuren. Auch die Ramdisk ist so aufgebaut. Da die Spuren als 16-Bit-Werte in einer Spurtabelle eingetragen sind und pro Spur zwei Verweise auf die Folgespuren existieren, kann die Tabelle maximal 256 Spuren aufnehmen. Die Ramdisk kann daher maximal 256*5 = 1280 KByte umfassen. Um nun die vergleichsweise 'riesige' Kapazität einer Festplatte von minimal 20MByte einigermassen vernünftig zu verwalten, mußte ich mir etwas einfallen lassen. Dabei hatte ich zu bedenken, daß das bisherige Dateiensystem aus Kompatibilitätsgründen erhalten bleiben sollte, daß die schon existierenden Programme unverändert laufen mußten und ich nicht unendlich lange entwickeln wollte. Dabei ist dann ein, wie ich meine, vernünftiger Kompromiß entstanden, der alle genannten Bedingungen erfüllt.

JADOS ermittelt die Kapazität der angeschlossenen Festplatte über ein Kommando des SCSI-Controllers und richtet dann automatisch sogenannte Partitionen ein. Jede Partition wird wie ein eigenständiges Laufwerk behandelt, sodaß auf einem physikalischen Medium mehrere voneinander unabhängige Logische existieren. Als Spurgröße habe ich 10 Sektoren eingerichtet. Bei 256 Spuren ergibt sich damit eine Kapazität von 2,5 MByte je logischem Laufwerk. Während die Disketten und die Ramdisk mit den Laufwerksbezeichnern "0".."4" angesprochen werden, habe ich für die Festplattenpartionen die noch freien Buchstaben "A"..."Z" verwendet. Da unser Alphbet 26 Buchstaben beinhaltet, kann JADOS 26*2.5MByte = 65MByte verwalten. Die folgende kleine Tabelle zeigt, welche Partionen bei einigen typischen SEAGATE-Platten eingerichtet werden:

ST225 (20 MByte netto)
—> Laufwerke "A" bis "H"

ST238 (30 MByte netto)
—> Laufwerke "A" bis "L"

ST251 (40 MByte netto)
—> Laufwerke "A" bis "P"

ST277 (65 MByte netto)
—> Laufwerke "A" bis "Z"

Die Anpassungen im Dateisystem waren minimal. Es mußten lediglich die neuen Laufwerksbezeichner akzeptiert werden und in der hierarchisch untersten Routine, die bis dahin schon in Disketten- und Ramlaufwerk unterschieden hatte, die Festplattenaufrufe eingebunden werden. Der große Vorteil dieser einfachen Plattenverwaltung besteht darin, daß die seit Jahren bewährten Unterprogramme des Dateisystems unverändert bleiben konnten. Sollte es dennoch einmal zu ernsten Störungen kommen, wird in der Regel nur eine Partition betroffen sein, während die anderen Partionen intakt bleiben.
Nach Einbau der Festplattenunterstützung habe ich die Gelegenheit genutzt und zahlreiche kleine Detailverbesserungen im JADOS eingebaut. Von besonderem Interesse sind die folgenden Features, die ich aus Platzgründen nur kurz anreißen kann:

Hardscroll

Erkennt JADOS, daß die GDP64HS im System ist, dann schaltet es auf den schnellsten Hardscrollmodus um. Falls noch die alte GDP64 vorhanden ist, bleibt nach wie vor der Softscroll aktiv

Editor

JADOS bietet mit dem Kommando "EDIT" die Möglichkeit, den im Grundprogramm enthaltenen Editor aufzurufen. Bisher konnte dieser nur mit Ctrl KX verlassen werden, worauf JADOS den Text auf den Massenspeicher zurückschrieb. Nun kann der Editor auch mit Ctrl KQ verlassen werden, wobei der Text nicht zurückgeschrieben wird. In manchen Situationen ist dies von großem Nutzen.

Grundprogramminterface

Da das Grundprogramm nun auch als Trap aufrufbar ist, nutzt JADOS diese Möglichkeit aus. Mit dem nun internen Kommando "GP" wird das Grundprogramm in der neuen Menüform aufgerufen. Es kann mit "Z" wieder verlassen werden, worauf man wieder im JADOS ist. Die Umschaltung zwischen JADOS und Grundprogramm geschieht damit blitzschnell und unproblematisch

Stepgeschwindigkeit
Viele Diskettenlaufwerke erlauben es, mit der minimalen Steprate von 3 ms zuarbeiten. Dadurch vermindert sich die durchschnittliche Zugriffszeit auf etwa 2/3. Außerdem wird dadurch der Geräuschpegel stark verringert. Mit dem neuen Kommando "STEP" kann die Steprate in acht Stufen zwischen 3 ms und 30 ms eingestellt werden. Der voreingestellte Wert beträgt die üblichen 6 ms

Assembler
Das Kommando "ASS" hat zusätzliche Qualitäten bei der Behandlung der Parameter erhalten. Wenn der erste Parameter das Zeichen "@" vorangestellt bekommt, so wird dies als automatische Antwortdatei ausgewertet, die eine Liste der Dateien enthält, die zusammen assembliert werden sollen. Der optional zweite Parameter bestimmt den Namen der Programmdatei. Es werden jetzt auch die Dateitypen .COM, .OBJ und .SYS akzeptiert. Alle anderen Endungen werden wie bisher als .68K interpretiert. Ein kleines Beispiel soll die Möglichkeiten des Kommandos deutlicher machen. Der Befehl:

```
ASS @DOSLINK JADOS.SYS
```

lädt alle in der Datei DOSLINK angegebenen Textdateien in den Speicher, assembliert und speichert das Programm unter dem Namen JADOS.SYS ab. Die Dateiendung .OBJ ist bereits im Hinblick auf den von Klaus Rumrich entwickelten Linker reserviert.

Wahlfreier Dateizugriff
Einige Programmierer bemängelten, daß eine Datei nur sequentiell gelesen oder beschrieben werden konnte. Mit der neuen Version steht nun ein Unterprogramm zur Verfügung, mit dem ein beliebiger Sektor einer Datei angewählt werden kann. Ein nachfolgender Lese- oder Schreibzugriff erfolgt dann auf den nächsten Sektor. Damit besteht endlich die Möglichkeit, wahlfrei auf Dateien zugreifen zu können.

Kommandointerpreter als Unterprogramm
Der Kommandointerpreter des JADOS kann über einen Trap aufgerufen werden.Damit ist es möglich, alle internen Kommandos sowie resident abgespeicherte Programme und 68K-Programme aufzurufen. Dieses Feature ist für Programmierer sehr wertvoll. Sinnvoll einsetzbar ist es allerdings nur in COM-Programmen, da nur dort der Bereich ab der Ladeadresse frei bleibt.

Konfigurationsdatei
Seit Version 3.0 gibt es einen Bereich am Anfang des JADOS, in dem einige Konstanten abgelegt sind. Mit diesen Konstanten werden bestimmte Parameter des JADOS eingestellt. Die Anzahl der Parameter wurde in der aktuellen Version stark erweitert. Da eine Änderung dieser Parameter durch den Benutzer fehleranfällig ist, habe ich den Urlader des JADOS so erweitert, daß er eine Textdatei namens CONFIG.SYS abarbeiten kann, die die Parameteränderungen beinhaltet. Im Klartext kann man z.B. eingeben auf welcher Adresse die Sound-Baugruppe liegen soll.

Als Besonderheit ist noch anzumerken, daß in der Konfigurationsdatei auch die Installation einer Ramdisk eingestellt werden kann. Die manchmal etwas lästige Frage des Urladers nach Einrichtung einer Ramdisk kann damit abgestellt werden. Nützlich sind auch die Angaben über das Vorhandensein oder Nichtvorhandensein der Diskettenlaufwerke. Die zeitaufwendige, automatische Prüfung kann dadurch umgangen werden.

Ausblicke in die Zukunft
Mit Version 3.5 läßt sich schon ziemlich professionell arbeiten. In Zukunft geplant sind ein hierarchisches Dateiverzeichnis und ein Batchprozessor, der programmähnliche Kontrollstrukturen abarbeiten kann

Ulrich Kracker

Teil 2

DSP zum 'reinschnuppern' mit dem NDR-Computer

Teil 2

Wie im ersten Teil dieser kleinen Abhandlung über das Digitale Signal Processing schon angedeutet, soll nun untersucht werden, ob und wie Signale auf ihre Frequenzanteile hin untersucht werden können. Dabei steht immer auch die Einbeziehung des NDR-Rechners und neuerdings auch des mic-Computers (im wesentlichen ein NDR-System auf ECB-Basis) im Vordergrund, denn wir wollen unserm 'guten Stück' etwas zum Futtern geben.

Daß mit einem entsprechend ausgerüsteten (NDR-) Rechner Signale aus der Umwelt erfaßt werden können, ist längst nichts Neues mehr. Der interessante Teil beginnt ja erst jetzt:
Wie kann man den eingelesenen Daten Informationen entnehmen, die einem beim bloßen Betrachten der zeitlichen Darstellung niemals auffallen würden?
Man könnte mit diesen Informationen einiges ergründen, z.B:

- Warum klingt ein Klavier anders als eine Pfeife, obwohl beide auf der gleichen Tonhöhe spielen?
- Wie bekommt man ein lästiges Pfeifgeräusch aus seiner Aufnahme heraus, ohne gleich alle Höhenanteile wegzudämpfen?
- Wie erzeugt man künstliche Klänge, wie sie in der modernen Musikelektronik gang und gäbe sind?

Das sind alles Fragen, die in die selbe Richtung weisen, nämlich:
Aus welchen Einzel-Komponenten ist ein beliebiges Probe-Signal aufgebaut?
Der Mathematiker Jean Baptiste Joseph Fourier (1768-1830) hat ein mathematisches Verfahren entwickelt, das es erlaubt auf diese Frage eine eindeutige Antwort zu finden. So weit, so gut! Die Zielsetzung ist klar. Wie sieht aber dieses Verfahren aus, wie setzt man es ein? Bevor der Einsatz eines solchen Verfahrens auf einem Rechner ins Auge gefaßt wird, kommt man nicht umhin, die berühmt-berüchtigten Grundlagen zu verinnerlichen. Nun denn...

Grundlagen zur Fouriersythese
Die *Fouriersyntese,* nach ihrem Entdecker

benannt, stellt einen Zusammenhang dar, zwischen einem beliebigen, mathematisch beschreibbaren Zeitsignal f(t) und den Frequenzkomponenten aus denen es sich rekombinieren läßt. Dieser Satz ist vielen sicherlich noch aus der Schulzeit bekannt; er besagt, daß beliebige Kurvenformen aus einer Ansammlung von Sinusschwingungen zusammengesetzt werden können. Dazu sind aber ein paar nicht unerhebliche Einschränkungen, oder besser, Bedingungen aufzustellen:

- Die einzelnen Sinusschwingungen werden mit dreiParametern festgelegt:
  * Amplitude c
  * Frequenz f
  * Phasenverschiebung _
- Die einzelnen Sinusschwingungen werden additiv miteinander verknüpft
- Die resultierende Kurvenform ist periodisch, d.h. sie wiederholt sich nach einer Periodenzeit T, ebenso wie diejenige Teil-Sinusschwingung, die die niedrigste Frequenz f = 1/T aufweist.

Der letzte Punkt erscheint vielleicht belanglos, wird aber später noch deutlich an Gewichtung zulegen.
Da zum Aufbau einer beliebigen Signalform mehrere, mitunter viele einzelne Sinusschwingungen benötigt werden, werden sie mit einem Index durchnummeriert. Der Index wird üblicherweise mit **n** benannt.
Die Frequenz f_n der einzelnen Schwingungen wird nicht etwa beliebig festgelegt, sondern in *festen Schritten* fortlaufend erhöht. Die Erhöhung der Frequenz f_n wird in ganzzahligen Vielfachen der tiefsten Teilfrequenz vorgenommen. Aus diesem Grund kann in Gleichung (6) gleich der Index n im Argument des Sinusausdruckes verwendet werden, um die Frequenz $f_n = n f_1$ zu bilden.
Die Teilschwingung mit der tiefsten Frequenz, auf der sich ja alle weiteren Teilschwingungen aufbauen, wird oftmals auch *Grundschwingung* genannt; sie erhält den Index n = 1 zugewiesen.
Im Fachjargon werden die auf der Grundschwingung aufbauenden Teilschwinhungen auch 'harmonische Schwingungen' oder verkürzt oft nur 'Harmonische' genannt.

In Gleichung (6) taucht aber auch noch der Index N = 0 auf. Damit wird eine spezielle Teilkomponente mit der Frequenz f = 0Hz bezeichnet, also ein additiver Gleichanteil.

Was hier nun mit vielen Worten zu erläutern versucht wurde, kann mit wenigen mathematischen Formeln exakt wiedergegeben werden:

$$f(t) = c_o + \sum_{n=1}^{N} c_n \sin(n f_1 2\pi t + \phi_n) \qquad (6)$$

mit:

$$c_n = \sqrt{(a_n^2 + b_n^2)} \qquad (7)$$

$$\phi_n = \arctan\frac{b_n}{a_n} \qquad (8)$$

$$a_n = \frac{2}{T_1} \int_{-\frac{1}{2}T_1}^{+\frac{1}{2}T_1} f(t)\cos(n f_1 2\pi t)dt \qquad (9)$$

$$b_n = \frac{2}{T_1} \int_{-\frac{1}{2}T_1}^{+\frac{1}{2}T_1} f(t)\sin(n f_1 2\pi t)dt \qquad (10)$$

$$a_o = \frac{1}{T_1} \int_{-\frac{1}{2}T_1}^{+\frac{1}{2}T_1} f(t)dt \qquad (11)$$

$$b_o = 0 \qquad (12)$$

$$T_1 = \frac{1}{f_1} \qquad (13)$$

Im obigen Gleichungsblock gibt es jedoch noch ein paar Dinge zu erläutern:

f(t) in Gleichung (6) repräsentiert das Zeitsignal, das über die Synthese aus den Einzelkomponenten gewonnen werden soll. Es handelt sich dabei um eine periodische, mathematisch beschreibbare Funktion, also zum Beispiel eine Sägezahnschwingung mit der Beschreibung:

$f(t) = u(t) = 25V/s\ t_{/_{-T<t<T}}$
(Sägezahnspannung)

In den Gleichungen (9) und (10) hingegen stellt **f(t)** das Zeitsignal dar, das in seine Bestandteile zerlegt werden soll.
Die Zeit T_1 ist der Kehrwert der Grundfrequenz f_1; er wird auch 'Periodenzeit der Grundschwingung' genannt.

Die *Fourieranalyse* erstreckt sich im Grunde auf die Ermittlung der beiden Integrale (9) und (10) für den Index n = 0...N. Die Größe N stellt dabei die Nummer der höchsten gewünschten Harmonischen dar und wird in der Regel so gewählt, daß die zugehörigen Oberwellenkoeffizienten vernachlässigbare kleine Werte aufweisen (Konvergierung vorausgesetzt).
Die Koeffizienten a_n und b_n, und damit auch die Koeffizienten c_n, lassen sich grundsätzlich jedoch nur bei periodischen Zeitsignalen f(t) berechnen, da die Zeit T_1 (Periodendauer der Grundschwingung) definiert sein muß, exakter, weil mathematisch ausgedrückt, lautet diese Forderung:

$$0 < T_1 < \text{unendlich} \qquad (14)$$

Wie analysiert man Sprache oder Musik?
In der Realität sind jedoch rein periodische Signale sehr selten anzutreffen. Viel häufiger trifft man auf Signale, die nicht periodisch verlaufen (z.B.: Sprachsignale). Das bedeutet nämlich, daß die Zeit T_1 gegen Unendlich strebt - Was Nun?!
Ein weiterer Pferdefuß bei der Integration nach Gleichung (9) und (10) ist, daß die zu analysierende Zeitfunktion f(t) mathematisch beschreibbar sein muß, d.h. sie muß wenigstens abschnittsweise bekannt sein. Was tun, wenn aber f(t) nicht bekannt ist, wie zum Beispiel beim obigen Sprach - oder Musiksignal?

Für diese Fälle gibt nun Tricks, auf denen die gesamte moderne Digitale Signalverarbeitung aufbaut.

Hier kommen wir nun auch an den Punkt, der eingangs schon einmal angedeutet wurde: Die Fouriersynthese, sowie auch die Analyse, haben strenggenommen nur für periodische Signale Gültigkeit!.
Es bleibt also nichts anderes übrig, als das unbekannte Meßsignal so hinzutrimmen, daß die Voraussetzungen geschaffen werden, um letztendlich doch die Fouriera-

nalyse darauf anwenden zu können.
Als erster Schritt in diese Richtung, wird ein ausreichend großer Zeitabschnitt von der Dauer T aus der nichtperiodischen Zeitfunktion f(t) herausgeschnitten. Von diesem Teilabschnitt wird von jetzt ab angenommen, er wäre periodisch mit der Zeitspanne T.

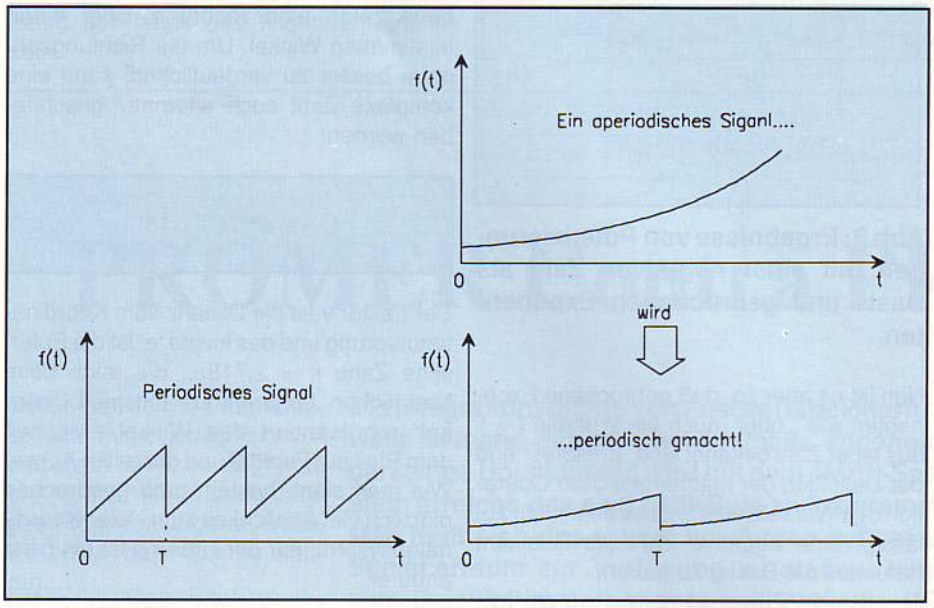


Abb. 6: Periodische und nichtperiodische Zeitsignale

Man erhält auch hier Teilschwingungen (=Harmonische) mit den Amplituden c_n und den Frequenzwerten:

$$f_n = n/T \qquad (15)$$

Wobei man sich vor Augen halten muß, daß die Zeit T_1 in den Gleichungen (9)...(13) sozusagen 'gewaltsam' gewonnen wurde, indem einfach ein Stück aus der Zeitachse der Länge T herausgebrochen wurde.
Durch die Wahl des Zeitraumes T kann also in gewissen Grenzen die Frequenzauflösung der Analyse (siehe Gl. (15)) variiert werden:

T groß : Frequenzrasterung fein
T klein : Frequenzrasterung grob.

Mit der Frquenzauflösung einhergehend, erhält man in beiden obigen Fällen unterschiedlich detaillierte Aussagen über die in f(t) enthaltenen Frequenzkomponenten, da im einen Fall mehr Information über den Verlauf von f(t) berücksichtigt wird (T groß) und im anderen Fall weniger (T klein).

Wenn die Zeit T gegen unendlich strebt, rücken die Frequenzwerte f_n immer dichter aneinander heran, bis aus der ursprünglichen (diskreten) Reihe eine (geschlossene) Funktion geworden ist. Man spricht dann von der Spektralfunktion C(f).

Die zweite Nuß, die zu knacken ist, ist die: Wie kann eine Funktion, die mathematisch nicht geschlossen beschreibbar ist, integriert werden (siehe Gleichungen (9) und (10))?
Ein möglicher Weg wäre es, mit Hilfe von Approximationsverfahren die Parameter einer bekannte Funktion f'(t) solange zu variieren, bis eine gute Übereinstimmung zwischen f'(t) und dem zufälligen Zeitsignal f(t) gefunden wird. Diese Methode ist jedoch relativ zeit - und rechenintensiv.

Eine zweite Methode ist die, daß von der Integration ...dt
abgegangen wird und statt dessen die Summenbildung $\sum$...dt eingeführt wird.
Dieser Schritt liegt deshalb nahe, da im Fall der computergesteuerten Signalerfassung die Funktion f(t) sowieso nicht in einer zeitlich lückenloser Form vorliegt, sondern vielmehr aus einzelnen Abtastwerten fk besteht, die immer um eine Abtastzeit T_A auseinanderliegen. Die Abtastzeit T_A wird dann zu obigen dt ernannt.
Das Stichwort lautet hierbei 'Numerische Integration'!
Diese Methode werden wir letztendlich auch für die praktische Anwendung der Fourieranalyse einsetzen.

Zuvor jedoch noch ein Zwischenschritt:

Übergang zur komplexen Form der Fouriertransformation:
Wozu, so wird sich der Leser fragen, muß jetzt auch noch auf eine komplexe Form eingegangen werden? Als ob die bisherige Theorie nicht schon komplex genug wäre!
Es wird sich später zeigen, daß durch eine komplexe Schreibweise der Fouriertransformation Vereinfachungen in Bezug auf obige Summenbildung möglich sind, die in der herkömmlichen Schreibweise nicht so offensichtlich sind. Ausserdem ist das Rechnen mit komplexen Zahlen ein elementarer Bestandteil der höheren Mathematik. Aus diesem Grunde wollen wir kurz die elementaren Zusammenhänge der komplexen Rechnung beleuchten:

Komplex ist nicht gleich kompliziert!
Wem das Arbeiten mit komplexen Zahlen und der Begriff 'Eulersche Indentität' ein Begriff ist, der kann getrost dieses Kapitel überspringen.
Viele unter den Lesern kennen sicherlich das 'Karthesische Koordinatensystem': Es ist die Anordnung zweier Maßstabsachsen im Rechten Winkel zueinander. In dieser

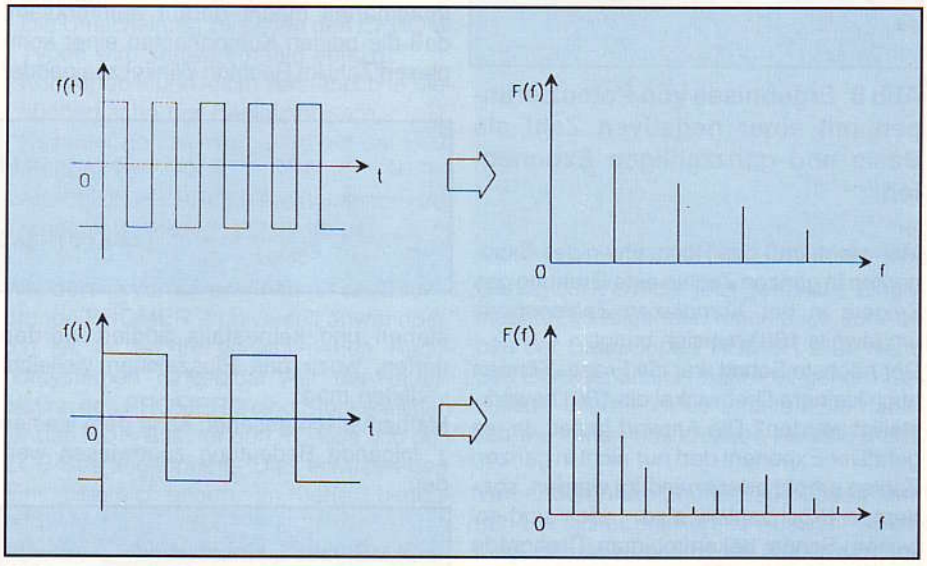


Abb. 7: Auswirkung der Periodenzeit T auf die Frequenzauflösung

zweidimensionalen Ebene lassen sich Punkte eintragen, die sich eindeutig über die Koordinatenangaben in x - und in y - Richtung wiederfinden lassen.
Daneben gibt es noch das sogenannte 'Polar-Koordinatensystem'. In dieser zweidimensionalen Ebene gibt es lediglich einen Zentralpunkt und von diesem ausgehend eine Bezugslinie in einer festgelegten Richtung. Eine beliebige Punkteangabe besteht auch hier aus zwei Teilen: Der Entfernungsangabe zum Zentralpunkt und einer Winkelangabe, bezogen auf die Bezugslinie.

Die komplexe Zahlenebene

Der Mathematiker Leonhard Euler (1707 - 1783) hat sich u. a. mit der Potenzierung (Begriffe: Basis und Exponent) in der Mathematik beschäftigt. Er fand heraus, daß negative Werte als Basis eingesetzt, besonders interessante Eigenschaften aufweisen. Diese Eigenschaften treten besonders dann hervor, wenn die Ergebnisse der Potenzierung in ein kartesisches Koorinatensystem eingetragen werden. Die Achsenbezeichnungen lauten 'relle Achse' und 'imaginäre - oder j-Achse'. Die darin eingetragenen Punkte werden durch Pfeile vom Urprung aus angezeigt.

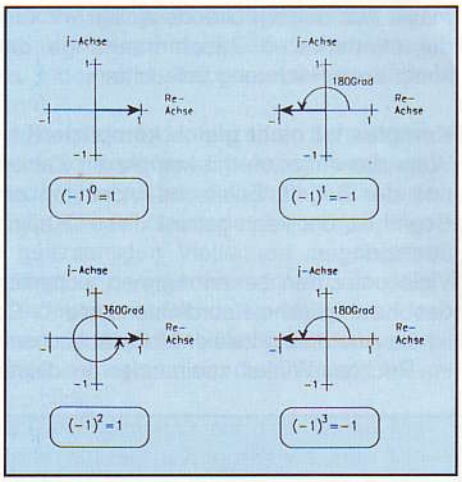


Abb 8: Ergebnisse von Potenzierungen mit einer negativen Zahl als Basis und ganzzahligen Exponenten.

Man sieht, daß das Hochzählen des Exponenten in ganzen Zahlen eine Drehung des Zeigers in der *Komplexen Zahlenebene* um jeweils 180° mit sich bringt.
Der nächste Schritt war die Frage: Können auch kleinere Drehwinkel als 180° bewerkstelligt werden? Die Antwort lautet: Ja, es geht!Der Exponent darf nur nicht in ganzen Zahlen erhöht oder erniedrigt werden, sondern in Bruchzahlen, also oder und so weiter. Schon bekommt man Drehgrade um 90° oder um 45°.

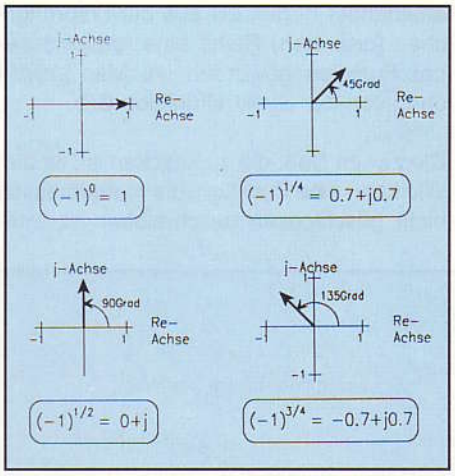


Abb 9: Ergebnisse von Potenzierungen mit einer negativen Zahl als Basis und gebrochenen Exponenten.

Nun ist es aber so, daß gebrochene Exponenten wie oder auch als Wurzeln (√) aus einer Zahl bekannt sind. Entsprechend der Definition der mathematischen Operation des Wurzelziehens ist es aber nicht zulässig, eine negative Zahl als Radikant (ehemals die Basis) zu haben.
Aus diesem Zwiespalt heraus wurde festgelegt, daß 'Koordinatenangaben', die nach dem Exponenten-Prinzip zustandekamen, folgendermaßen deklariert werden:

$$\underline{C} = y\ (\text{Realteil} + \mathbf{j}\ \text{Imaginärteil}) \qquad (16)$$

C wird dabei als **'komplexe Zahl C'** bezeichnet.
Die beiden Achsenabschnitte: Realteil und Imaginärteil geben letzlich die Richtung des zur komplexen Zahl gehörenden Zeigers an. Der Wert y ist lediglich ein Maßstabsfaktor. Das ominöse kleine j vor dem Imaginärteil macht darauf aufmerksam, daß die beiden Komponenten einer komplexen Zahl im Rechten Winkel zueinander stehen und keinesfalls addiert werden dürfen, wozu das Pluszeichen vielleicht verleiten mag.
Mathematisch gesehen kann dem kleinen 'j' folgende Bedeutung zugewiesen werden:

$$j = \sqrt{(-1)} \qquad (17)$$

Die Wurzel aus einem negativen Radikant ist bekanntlich nicht definiert (mathematische Umschreibung für: 'geht nicht'). Aus diesem Grund läßt man den Ausdruck einfach so stehen, gibt ihm einen Namen (j) und rechnet damit weiter.
Wie angedeutet, beschreiben Realteil und Imaginärteil eines komplexen Zeigers, der auf einen Punkt in der komplexen Zahlenebene zeigt, eine Richtung, oder einen bestimmten Winkel. Um die Richtungsangabe besser zu verdeutlichen, kann eine komplexe Zahl auch alternativ geschrieben werden:

$$\underline{C} = y\, e^{j\text{winkel}} \qquad (18)$$

Der Faktor **y** ist die Distanz vom Koordinatenursprung und das kleine 'e' ist die Eulersche Zahl: e = 2,718..., die auch beim Natürlichen Logarithmus mitspielt. **'winkel'** repräsentiert den Winkel zwischen dem Pfeil auf Punkt C und der rellen Achse.
Wie man sieht, besteht grob gesprochen plötzlich viel Ähnlichkeit zum Polar-Koordinatensystem, nur der Urprung ist ein ganz anderer.

Was bringt die komplexe Ebene?

Mit einer komplexen Zahl lassen sich zwei Komponenten auf einmal erfassen, daher auch der Name 'komplex'.
Wie in der Trigonomie gibt es auch hier eine Periodizität (=Wiederholung von Funktionswerten), denn eine Potenzierung von (-1) mit 2 oder mit 4 ergibt wieder den selben Wert, wie die Potenzierung mit 0.
In der Tat hat der Mathematiker Euler eine Beziehung gefunden, die es erlaubt, eine komplexe Zahl in einen (imaginären) Sinusteil und einen (reellen) Cosinusteil aufzuspalten. Damit kommt man wieder auf die ursprünglich dargestellte Schreibweise zurück:

Die Eulerschen Beziehungen:

$$\underline{C} = C\, e^{j\Phi} = C\,(\cos(\phi)+j\sin(\phi)) = \frac{a + jb}{2} \qquad (19)$$

$$\underline{C}^* = C\, e^{j(-\Phi)} = C\,(\cos(\phi)-j\sin(\phi)) = \frac{a - jb}{2} \qquad (20)$$

C* bedeutet 'konjungiert komplexe Zahl', und soll ausdrücken, daß der Winkel des komplexen Zeigers C jetzt ein negatives Vorzeichen erhalten hat. Dies hat zur Folge, daß dessen Winkel im Uhrzeigersinn gezählt wird, also einen entgegengesetzten Drehsinn zu C aufweist. In der Mathematik und in der Physik kann man diesen Umstand gelegentlich sehr elegant aus-

nützen. Hier sei es nur der Vollständigkeit halber aufgeführt.
Mit diesem Wissen läßt sich nun der Bogen zurück spannen, zum Problem der Fouriertransformation.
Doch damit sei es für dieses Mal genug der Theorie. Wie nun die Fouriertranformation auf einem Rechner und insbesondere auf dem NDR - Rechner übertragen werden kann, soll Inhalt des folgenden Artikels sein.

Literatur:

Bartsch: Taschenbuch Mathematischer Formeln, Verlag Harri Deutsch, 1984

Norbert Schäfer, Manfred Bertuch: 'Butterfly-Algorithmus', erschienen in c't 8/86, Verlag Heise

Dr. R. Best: 'Digitale Meßwertverarbeitung' erschienen in der Reihe: tm - Gastvorlesung, Jahrgänge 1988...90. Verlag Oldenbourg

Thomas Kieninger

PROMER Contra PROMER 2

Die Integrationsdichte von Festwertspeichern hat mittlerweile beinahe gigantische Ausmaße angenommen. Der derzeitige Höchststand wird mit dem 4MBit-Speicherbaustein erreicht, das sind 512KByte an Programmspeicher! Man muß sich das einmal vor Augen halten: Um diesen Speicherraum zur Verfügung zu stellen, benötigt man zum Beispiel immerhin acht vollbestückte ROA64-Baugruppen (je 64KByte) im NDR-Rechner.

Bevor man solche Programmspeicherriesen jedoch sinnvoll einsetzen kann, benötigt man erst einmal ein leistungsfähiges Programmiergerät, wobei wir schon beim Thema währen...
Bevor der neue PROMER 2 näher vorgestellt wird, soll ein kurzer Rückblick auf seine geschichtliche Entwicklung eingeschoben werden.
Die altbewährte NDR-Baugruppe 'PROMER' ist ja schon seit einiger Zeit auf dem Markt. Sie war im wesentlichen auf die im NDR-Sytem einsetzbaren EPROM-Bausteine abgestimmt: 2716, 2732 und 2764. Die Konzeptionierung des PROMERs war auf einfachen Schaltungsaufwand und damit auf ein leichtes Verständnis ausgelegt. Dennoch war das Arbeiten damit für damalige Verhältnisse (1984) komfortabel und die Ergebnisse zuverlässig.
Die Auswahl der EPROMs erfolgt über Kodierstecker, d.h. man benötigt speziell verdrahtete DIL-Stecker für jeden EPROM-Typ. Die Programmierspannung muß extern, durch ein zusätzliches Netzteil das die benötigten Spannungen 25V, 21V und 12,5V erzeugt, auf die EPROMS gegeben werden. Außerdem konnte die Baugruppe PROMER nur im NDR-Computer eingesetzt werden. Als ein weiterer Nachteil dieser Baugruppe hat sich die Erzeugung der Programmierzeit herausgestellt: Sie wurde durch ein RC-Monoflop gelöst. Dies erforderte erstens einen Abgleich und zweitens ist ein solches RC-Glied natürlich temperatur- und alterungsabhänig, so daß ein Nachregeln unter Umständen erforderlich wurde.
Wie gesagt: Diese Lösung war für "gestandene" NDR-Anwender durchaus tragbar, das belegen die abgesetzten Stückzahlen eindeutig. Da nun aber auch NDR-Baugruppen entwickelt wurden, auf denen sich auch höherintegrierte EPROMs einsetzen lassen, war der Ruf einer Neuentwicklung unüberhörbar.

Voilà: Dies sind die Eigenschaften der Baugruppe PROMER 2:

- Universeller Einsatzbereich (NDR, ECB und PC)
- Alle benötigten Programmierspannungen werden auf der Baugruppe selbst erzeugt.
- Die Umschaltung zwischen den EPROM-Typen wird softwaremäßig erledigt.
- Komfortable und leicht verständliche Menüoberfläche der Begleitsoftware.
- Keinerlei Abgleichvorgänge auf der Bau gruppe.
- Alle üblichen EPROM-Typen können programmiert werden. (2716...27011)

Wie bereits vorher erwähnt, ist die Baugruppe PROMER 2 universell anwendbar, d.h. sie ist auf drei verschiedenen Rechnersystemen einsetzbar. Auf der Hauptplatine des PROMER 2 sind Steckerleisten für den NDR-Bus, für den PC-Bus und den ECB-Bus angebracht. Dies erweitert den Einsatzbereich enorm, im Klartext bedeutet dies:
PC-Bus - Einsetzbar in jedem IBM kompatiblen PC, XT oder AT Computer mit dem Betriebssystem MS-DOS oder PC-DOS.
NDR-Bus - Einsetzbar im NDR-Computer mit all seinen Ausbaustufen, Z80 + CP/M2.2 oder 680XX + neuem Grundprogramm V6.2.
ECB-Bus - Einsetzbar im mc-CP/M Computer und im neuen **mic**-System.
Hier noch eine Aufstellung der EPROM-Typen die die Baugruppe PROMER 2 programmieren kann.

EPROM -Typ	Programmierspannung	Speichergröße
2716	25V	16 KBit
2732	25V	32 KBit
2764	12,5V oder 21V je nach Typ	64 KBit
127128	12,5V oder 21V je nach Typ	128 KBit
27256	12,5V oder 21V je nach Typ	256 KBit
27512	12,5V	512 KBit
27513	12,5V	512 KBit
27010	12,5V	1024 KBit
27011	12,5V	1024 KBit

Genug der Vorrede, jetzt gehts ans Eingemachte: Im folgenden möchte ich auf Aufbau der Baugruppe PROMER 2 an Hand des Blockschaltbilds näher eingehen. Detailierte Informationen erhalten Sie natürlich wie immer aus unseren Handbüchern.

Von anderen Peripherie-Baugruppen kennen Sie sicher die Adress-Auswahl-Schaltung, prinzipiell besitzt sie immer den selben Aufbau und dieselbe Funktionsweise.

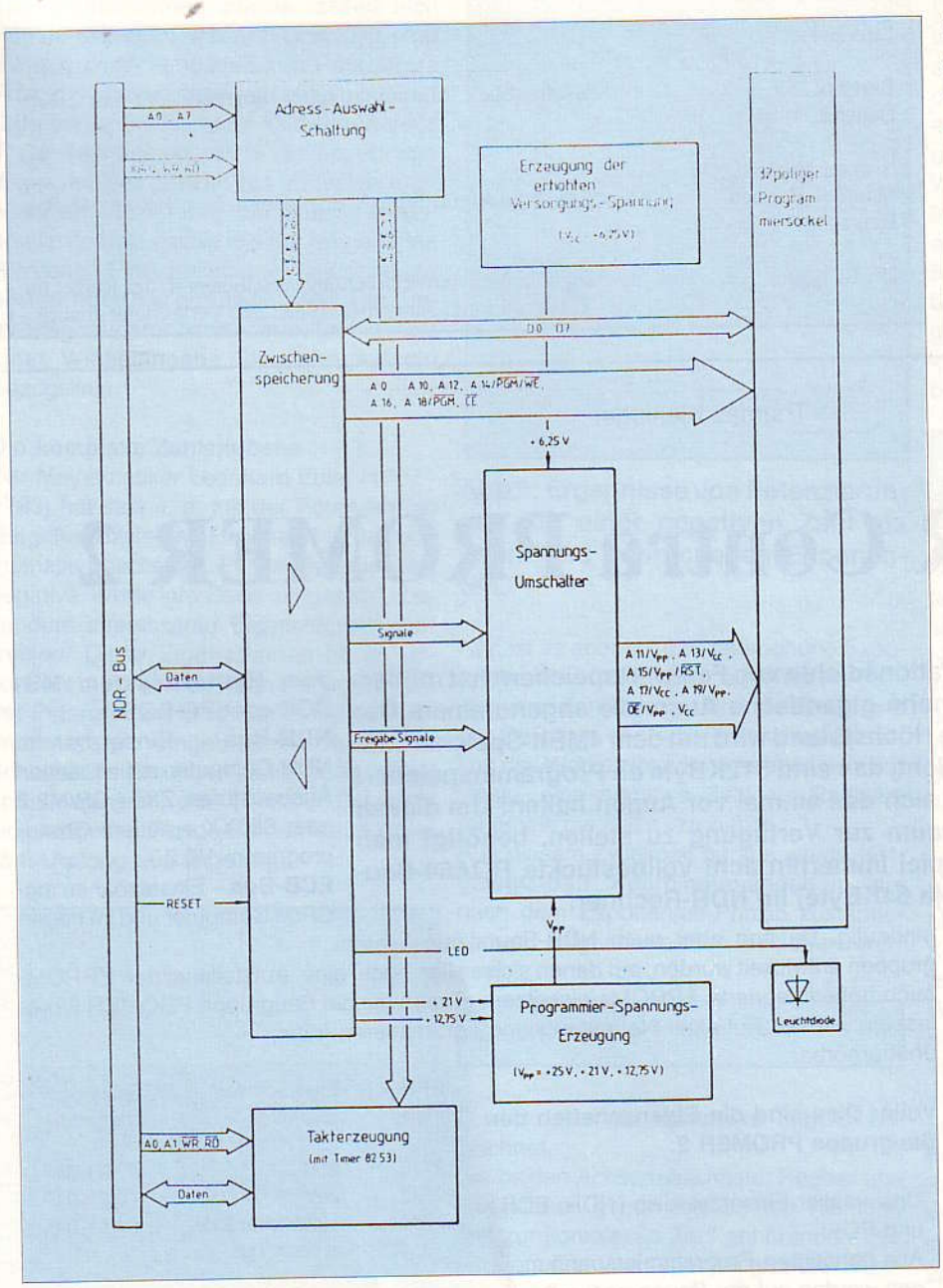


Abb. 1: Blockschaltbild

Bei der Ansteuerung einer Peripheriekarte im Computers erfolgt die Kommunikation zwischen dem Prozessor und der Baugruppe über den Datenbus des Rechners. Hierbei wird beim Schreibvorgang auf eine I/O-Baugruppe ein separates Steuersignal aktiviert, um eine Unterscheidung von einem Speicherzugriff vornehmen zu können. Jede Baugruppe bekommt eine oder mehrere Adressen zugeordnet. Solch eine I/O-Adresse wird als "Port" bezeichnet.

Für die Baugruppe PROMER 2 stehen acht Portadressen zur Verfügung. Je Port muß sowohl ein Schreib- als auch ein Lesevorgang möglich sein. Es ist daher die Erzeugung von 16 Port-Auswahl-Signalen erforderlich. Die Dekodierung der Port-Adressen und die Erzeugung der Port-Auswahl-Signale übernimmt die Adress-Auswahl-Schaltung.

Es läßt sich sicher einsehen, daß es notwendig ist, die Daten und Adress-Signale während des eigentlichen Programmiervorganges festzuhalten, für diesen Zeitraum also zwischenzuspeichern. Hierfür werden sogenannte Latches verwendet. Es handelt sich um acht D-Flip-Flop, die einen gemeinsamen Takt-Eingang besitzen. Der Datenbus des Rechners wird an die Eingänge der D-Flip-Flops gelegt und das zugehörige Port-Auswahl-Signal, welches vorher in der Adress-Auswahl-Schaltung produziert wurde, als Takt verwendet.

Die benötigten Programmierspannungen 25V, 21V, 12,5V und die erhöhte Versorgungsspannung von 6,25V der EPROMs werden auf der Baugruppe selbst erzeugt. Die einzige Möglichkeit, die geforderte Spannung zu erzeugen, liegt darin, diese Spannungen aus den vorhandenen Gleichspannungen +5V oder +12V gewinnen. Dies wird durch einen Aufwärtsregler realisiert.

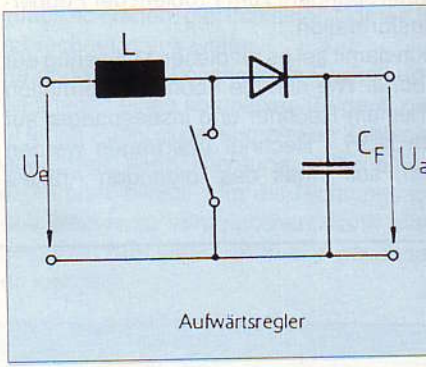


Abb. 2: Aufwärtsregler

Obenstehende Zeichnung zeigt den prinzipiellen Aufbau eines Aufwärtsreglers. Wird der Schalter geschlossen, baut sich ein Strom durch die Induktivität auf. Dieser Stromfluß steigt um so mehr an, je länger der Schalter geschlossen bleibt. Wird nun der Schalter wieder geöffnet, muß der Induktivitätsstrom wieder abgebaut werden. Dies geschieht, indem er durch die Diode in den Kondensator fließt. Am Kondensator verursacht aber eine Stromänderung auch eine Spannungsänderung. Die Spannung am Kondensator steigt also an. Es wird förmlich mit jedem Schließen und Öffnen des Schalters Ladung in den Kondensator hineingeschaufelt. Die Höhe der entstehenden Spannung hängt also von der relativen Ein- und Ausschaltdauer dieses Schalters ab. Durch solche Aufwärtsregler werden in der PROMER 2 Baugruppe die erhöhten Programmierspannungen und die erhöhte Versorgungsspannung generiert.

Da mit dieser Baugruppe soviele verschiedenen EPROMS programmiert werden können und jedes EPROM eine andere Belegung aufweist, bleibt es nicht aus, daß die Pins des Programmiersockels mehrfach belegt sind, d.h. es liegen unterschiedliche Signale (TTL-Pegel, Programmierspannungen, erhöhte Versorgungsspannungen) entsprechend der verwendeten EPROMs an den Pins an. Um die sogenannte Spannungsumschalter kann diese Umschaltung zwischen TTL-Pegel, den Programmierspannungen 12,5V, 21V, 25V und der erhöhten Versorgungsspannung 6,25V unter Softwarekontrolle vorgenommen werden, was den Bedienkomfort beträchtlich erhöht.

Die verschiedenen EPROM-Typen benötigen ausserdem unterschiedlich lange Programmierzeiten. Diese Zeit kann je nach EPROM-Typ 100us, 1ms oder 50ms betragen. Durch einen speziellen Schaltkreis (

von Intel), der intern drei unabhänig voneinander arbeitende Zähler besitzt ist es möglich diese drei verschiedenen Programmierzeiten unabhänig vom Systemtakt 'quarzgenau' zu erzeugen.
Das waren in kurzen Zügen die grundsätzlichen Funktions-Blöcke der Baugruppe PROMER 2.
Daraus wird ersichtlich, daß ein relativ großer technischer Aufwand betrieben wurde, um einen EPROM-Programmierer zu realisieren, der den heutigen Anforderungen bezüglich der Ergonomie und der Zuverlässigkeit voll und ganz erfüllt.
Natürlich muß auch eine adequate Software bereit stehen, um den PROMER 2 voll auszunutzen. In einer der nächsten LOOP wird eine PROMER- Bedieneroberfläche vorgestellt, die sich gewaschen hat! Als Stichpunkte sollen Window- Oberfläche und SAA-Standard genannt werden.

Roman Kircher

Unterbrechungsfreie Stromversorgung für PC's

Über Datensicherheit und - Sicherung wird heute viel gesprochen und es gibt auch alle möglichen Einrichtungen, die dazu angeboten werden, wie Streamer, schnelle Backup-Programme und sonstige Software. Bei allen auftretenden Störungen wird aber das Stromnetz als Fehlerquelle meist stark vernachlässigt.

Treten bei Ihnen auch immer wieder uner-klärbare Störungen am Rechner auf? Programme stürzen ab und wichtige Daten werden verstümmelt oder sogar zerstört?
In vielen solcher Fälle ist der Fehler nicht am Rechner oder bei der Software zu suchen, sondern im Stromnetz. Unser Stromnetz gehört zwar weltweit zu den besten, trotzdem kommt es statistisch gesehen 2-7 mal im Monat zu Stromausfällen. Diese Ausfälle sind zwar meist von kurzer Dauer - Bruchteile von Sekunden bis hin zu wenigen Minuten - , aber sie reichen aus, um einen PC zum Absturz zu bringen.

Wer oder was kann denn Stromunterbrechungen oder Störungen verursachen?
Die bekanntesten Störenfriede sind wohl die Gewitter. Bei Spannungsspitzen von einigen zehntausend Volt ist es kein Wunder, daß so manche Freileitung ihren Teil abbekommt.
Aber auch der Nachbar, der einen starken Elektromotor oder andere, nicht sauber entstörte große und kleine Elektrogeräte - auch Haushaltsgeräte - anschaltet, kann bei Ihnen eine Störung bewirken, die Sie gar nicht merken; aber der Computer merkt sie ganz deutlich.

Aber was passiert den nun mit meinem PC, wenn der Strom plötzlich weg ist?
Der Computer ist mit Disketten, Festplatten und andern Daten-speichern ausgerüstet, auf denen man Daten auf Dauer sichern kann. Ein Großteil der Arbeit wird aber im Hauptspeicher des Rechners geleistet, und dieser Hauptspeicher ist leider sehr "flüchtig". Es reichen schon einige tausendstel Sekunden, und die Daten sind ver-loren. 5-10 Millisekunden (tausendstel-Sekunden) kann das Netzteil des Computers überbrücken, aber ist der Strom länger weg, stürzt der Rechner hoffnungslos ab.
Vielleicht haben Sie das schon selbst erlebt: ein kurzes Flackern der Lichter bei einem auch entfernten Gewitter, und am Bildschirm tut sich gar nichts mehr. Im Fachjargon heißt das, der Computer hat sich "aufgehängt."
Wenn der Rechner in diesem Moment gerade nicht gearbeitet hat - er stand zum Beispiel im DOS bei [C:>] oder im Hauptmenü Ihres Programms - , ist das nicht weiter schlimm, denn alle Daten sind abgespeichert und somit gesichert.
Schlimmer ist schon der nächste Fall: Sie haben gerade einen 2 oder 3 Seiten langen Brief geschrieben und wollen Ihn gerade abspeichern. In diesem Fall dürfen Sie die 2 Seiten noch einmal tippen.
Das nächstgrößere Problem ist der Stromausfall während eines Programmlaufes (Compiler, Faktura, CAD-Berechnung, Abbuchungen...). Hier kann es je nach Anwendung vorkommen, daß Sie den unterbrochenen Programmablauf nicht mehr ohne weiteres neu starten können, ohne Daten zu verfälschen.
Der schlimmste Fehler ist der, daß teure Hardware zerstört wird. Jeder vernünftige Computertechniker predigt seinem Kunden, den Computer ja nie auszuschalten, während das "rote Licht" der Festplatte oder des Diskettenlaufwerks noch brennt. Wenn alles glatt geht, passiert nichts, aber im ungünstigsten Fall kann die Festplatte einen "Hardware-" Fehler davontragen. Das kommt zwar beim heutigen Stand der Technik kaum mehr vor, aber Ausnahmen bestätigen die Regel.

Fazit des Ganzen: Sie haben mit Ihrem PC keine Chance, wenn der Strom ausfällt (auch nur kurzfristig).
Aber findige Leute haben natürlich auch dafür ein schlaues Gerät entwickelt: eine USV - Unterbrechungsfreie Stromversorgung.
Ein solches Gerät ist natürlich kein Kraftwerk und Sie können damit nicht Ihre Wohnung heizen, aber es kann einen PC 10 Minuten oder auch länger mit Strom versorgen.

Wie funktioniert das Ganze nun?
Das Gerät wird ganz normal an das Stromnetz angeschlossen. Eine elektronische Schaltung überwacht ständig die Netzspannung. Fällt diese auch nur für den Bruchteil einer Sekunde aus, schaltet das Gerät innerhalb von 2-4 Millisekunden auf Notbetrieb um. Das heißt, ein im Gerät vorhandener Akku speist einen "Wechselrichter", der aus dem Gleichstrom der Batterie 220 Volt Wechselstrom für Ihren Computer macht. Das Umschalten geschieht dabei so schnell, daß Ihr Rechner völlig überrumpelt wird und nichts davon merkt. Er arbeitet einfach weiter.
Je nach Kapazität des Akkus und der Leistung der angeschlossenen
Geräte reicht der Strom 10 - 60 Minuten. Auf jeden Fall aber solange, daß Sie Ihre Arbeit ordnungsgemäß beenden und Ihre wichtigen Daten sichern können.
Ein eingebauter Schutzmechanismus verhindert ein "Tiefentladen" des Akkus und schaltet den Strom bei zu schwacher Batterie nach vorheriger Warnung ab.
Aber die Erfahrung zeigt, daß Netzausfälle in der Regel so kurz sind, daß die Akkuladung bis zum Wiedereintreffen des Stroms ausreicht.
Der Akku wird von der Elektronik im Gerät natürlich wieder automatisch aufgeladen und ist im übrigen wartungsfrei.

Mit einem solchen "schlauen" Gerät haben Sie alle langen oder kurzen Stromunterbrechungen im Griff.
Sollten Sie in einer Gegend wohnen, wo das Netz großen Stromschwankungen (keine Unterbrechungen) ausgesetzt ist, kann Ihnen mit einem Spannungskonstanthalter geholfen werden. Ein solches Gerät "bügelt" alle Schwankungen im Netz zusätzlich aus und eine Kombination mit einer USV-Anlage sichert Sie elektrisch so ab, daß Ihrem Computer nichts mehr passieren kann.

Für Novell-Netzwerk-Benutzer gibt es noch einen schönen Zusatz zur USV. Das Gerät kann mit potentialfreien Ausgängen ausgerüstet werden und liefert dem Novell-Server über eine kleine Zusatzkarte die Information "Strom aus". Nach einer vom Betreiber definierten Zeit und vorheriger Mitteilung an eventuell noch laufende Arbeits-stationen werden alle offenen Date
en des Servers geschlossen, der Inhalt d
RAM-Buffer auf die Platte geschrieben ur
ein vorschriftsmäßiger "DOWN" durchg
führt. Sollte die Akkuladung vorher z
Ende gehen, wird der "DOWN" von eine
weiteren Signal aus der USV erzwunger

Für genauere technische Auskünfte ur
andere Informationen steht Ihnen uns
Team jederzeit gerne zur Verfügung.

Bruno Sontheim

Analoge und digitale Regler-Simulation mit RESI.

Die folgende Beschreibung geht davon aus, daß der Leser mit den Grundbegriffen der Regelung vertraut ist. Nichts desto trotz ist das Programm RESI gerade für den Einsteiger in die Regelungstechnik gedacht, da ihm damit die Möglichkeit gegeben wird die Vorgänge im Regelkreis zu verstehen und beliebige Regelkreiskonstruktionen nachzubilden und zu testen. Ein entsprechender Lehrgang 'Einführung in die Regelungstechnik' welcher auf dem Programm RESI aufbaut, ist in Vorbereitung.

Die Regelungstechnik umgibt uns in fast allen Lebensbereichen. Wie man bei oberflächlicher Betrachtung meinen könnte, ist die Regelungstechnik keineswegs nur auf die Technik beschränkt. Sie ist als Hilfswissenschaft zum Beispiel auch in der Biologie und Soziologie zu finden. Man denke nur an die komplexesten Regelfunktionen die im menschlichen Körper zu finden sind. Der Mensch war immer bemüht sich die Natur und ihre Eigenschaften zunutze zu machen. So auch in der Regelungstechnik.
Die Planung komplexer technischer Prozesse kann durch die Simulation wesentlich vereinfacht werden. Teil-Prozesse können so vor der Fertigstellung des Anlagenteils optimiert werden. Planungsfehler können in einem frühen Entwicklungsstadium erkannt und damit preiswert korrigiert werden. Das Zusammenspiel der Teilprozesse kann ohne Gefahr in kritischen Situationen getestet und analysiert werden. Natürlich sind nicht alle Einflüsse mit einem Simulationsprogramm darstellbar, aber Simulationsprogramme sind aus der modernen Technik nicht mehr wegzudenken.
Das Simulationsprogramm RESI ist speziell für den Anfänger gedacht, der die Vorgänge im Regelkreis begreifen und analysieren lernen soll.

Das Programm RESI

Das hier beschriebene Programm RESI erlaubt die Simulation von Eingrößenregelungen. Unter einer Eingrößenregelung versteht man einen Regelkreis welcher eine physikalische Größe mit Hilfe eines Meßwertes regelt. Eingrößenregelungen sind in ungezählten Anwendungen zu finden:

Hier einige Beispiele:
- Geschwindigkeitsregelung eines Casettenrecorder-Motors
- Temperaturregelung im Backofen
- Automatische Fokusierung eines Objektivs im Fotoapparat
- Spannungsregler in Netzteilen

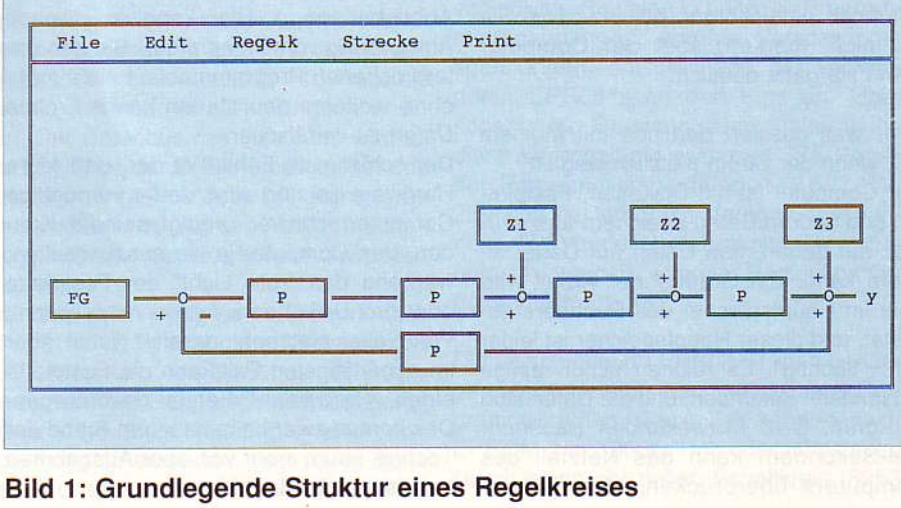


Bild 1: Grundlegende Struktur eines Regelkreises

Das Programm RESI ist
zwei große Teile gegliedert:
- Editierung eines Regelkrei
  ses und File
- Berechnung der Simulatio
  und graphische Darstellun
  der Regelkreisgrößen.

Editierung und Fileverwa
tung

Jeder Regelkreis hat die gle
che Struktur (Bild 1). Im Edit-Modus kö
nen den einzelnen Blöcken Parameter z
geordnet werden (Bild 2). Hier ist zum Be
spiel die Eingabe der Führungsgröße da
gestellt.

Folgende Einstellungen der Führungsgr
ße und Störgrößen können gemacht we
den:
- Signalart : Sinus, Rechteck und Ramp
  (Dreieck)

- Frequenz : beliebig
- Amplitude : -10..+10 (dimensionslos)
- Offset : -1..+10 (dimensionslos)

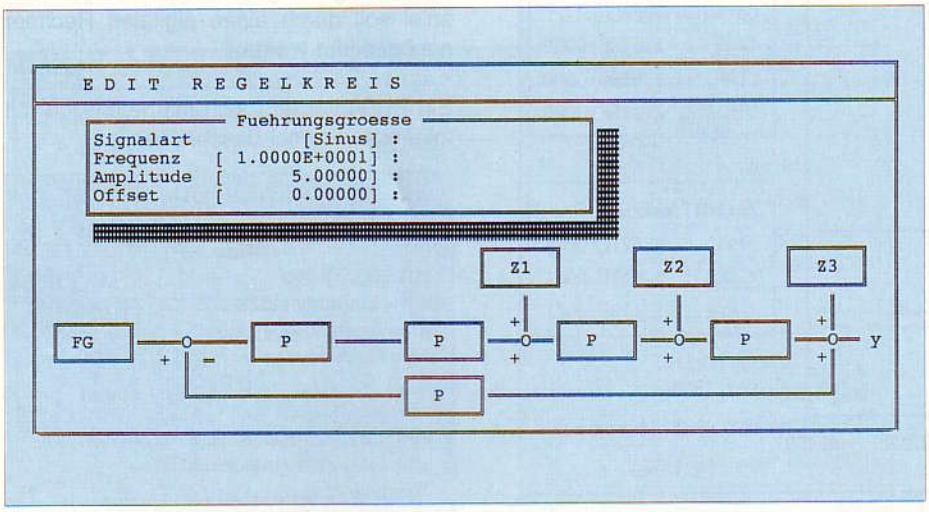


Bild 2: Menügeführtes Einstellen der Parameter

Die Regler-Block-Parameter werden mit dem gleichen Schema eingegeben. Es können folgende Reglertypen ausgewählt werden:

analog : P, PD, I, PI, PID, Zweipuntkregler und Dreipunktregler
digital : multiplizierend, rekursiv und Kompensationsregler

Bei Wahl einer digitalen Regelung muß zusätzlich die Abtastzeit des Abtast-Halte-Gliedes eingegeben werden (siehe auch Grundlagenartikel: DSP zum reinschnuppern, Teil 1 in LOOP 23).
Da man in der Technik meistens keine Regelstrecken (unter Regelstrecke versteht man den Teil des Regelkreises in dem die Regelgröße konstant gehalten werden soll; als Beispiel gelte der Backofen dessen Temperatur mit Hilfe der Regelung auf einer bestimmten Temperatur bleibt) findet die mit einer definierten Formel beschrieben werden können, wurde hier die Möglichkeit geschaffen 3 verschiedene Regelstrecken hinereinander zu schalten. Auch das Meßglied welches die Regelgröße für den Regler bereitstellt ist nicht immer ein Proportionalglied ohne Zeitverzögerungen. Auch hier ist die Möglichkeit gegeben verschiedene Eigenschaften nachzubilden.
Den Regelstrecken sowie dem Meßglied können folgende Streckenarten zugewiesen werden: P, PT1, PT2, I, LAU
Nach der Anwahl eines Streckentyps werden automatisch nur die in Betracht kommenden Variablen zur Parametrierung angeboten. Fehleingaben und unerlaubte Eingaben werden zurückgewiesen oder auf einen gültigen Wert auf- oder abgerundet.

Editierte Regelkreise können auf Diskette abgelegt und wieder geladen werden. Ein Daten-Directory kann definiert werden. Die abgespeicherten Regelkreise können im aktuellen Directory alphabetisch sortiert auf dem Bildschirm angezeigt werden. Bild (3) zeigt das Filemenü mit geöffnetem Daten-Fenster.

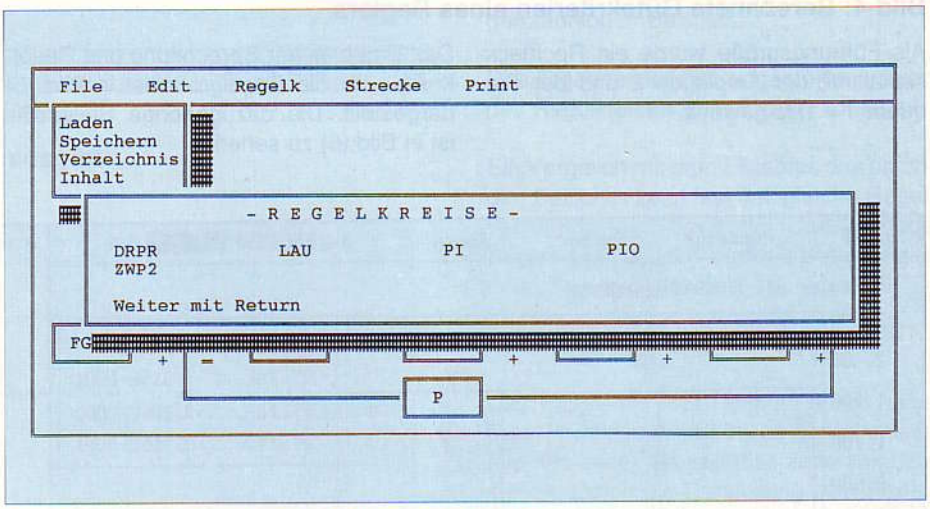


Bild 3: Menügeführtes Arbeiten mit Dateien

Berechnung und graphische Darstellung

Der Regelkreis wird mit Hilfe der numerischen Lösung von Differentialgleichungen erster Ordnung berechnet. Die Schrittwahl richtet sich nach der kleinsten vorkommenden Zeitkonstante im Regelkreis. Dies ist erforderlich um keine allzugroßen Genauigkeitsverluste in der Berechnung zu erhalten.

Als Hilfsmittel zur Optimierung von Regelkreisen werden 3 verschiedene Gütekriterien berechnet. Die Ergebnisse werden sofort nach der graphischen Ausgabe der aktivierten Regelkreisgrößen ausgegeben (Bild 4).

Die Gütekriterien sind:

- ITAE Integral of Time multiplied Absolute value
- LIRF Betrag der linearen Regelfläche
- QIK Quadratische Regelfläche

Die Zeitkoordinate des Darstellungsfensters wird automatisch vom Programm berechnet, kann aber beliebig verändert werden. Einmal berechnete Kurven können ausschnittsweise vergrößert oder auch verkleinert dargestellt werden.
Da die Sprungantwort in der Beurteilung von Systemen eine fundamentale Bedeutung in der Regelungstechnik besitzt, kann die Berechnung für den Regelkreis sowie für die einzelnen Strecken gewählt werden:

a) Führungsantwort
b) Sprungantwort

Bei der Führungsantwort wird der Eingang des Regelkreises mit dem Führungsgrößengenerator, bei der Sprungantwort wird dem Regelkreis bzw. der Strecke mit einem 'Sprung' gespeist.
Zur Messung der verschiedenen Größen stehen 2 Cursor zur Verfügung die mit den Pfeiltasten der Tastatur positioniert werden können. Zugleich werden die Amplituden der aktiven Regelkreisgrößen angezeigt. Aktive Regelkreisgröße bedeutet, daß diese Größe für die Anzeige aktiviert ist. Zur (De-) Aktivierung dient der Menüpunkt Anzeige.

Beispiel eines analogen Reglers

Es soll eine PT1-Strecke mit einem PI-Regler geregelt werden:
Die PT1-Strecke hat folgende Parameter:
Ks = 1.2
T1 = 0.1 sec
Gewählte Parameter für den Regler:
Kr = 2
Tn = 0.025 sec

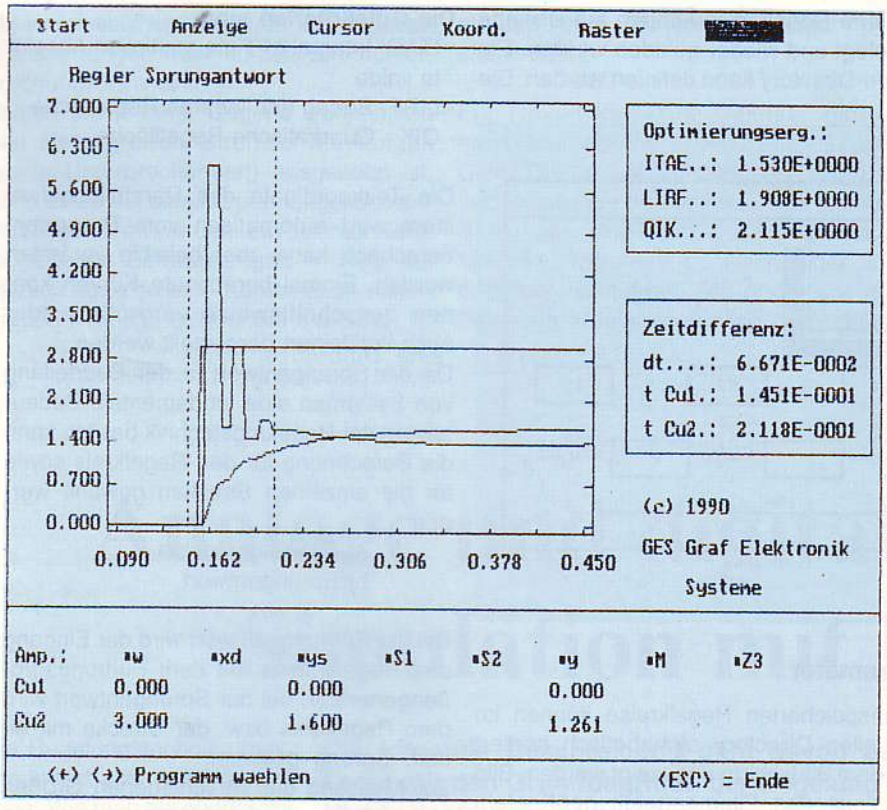


Bild 4: Berechnete Gütekriterien eines Reglers

Als Führungsgröße wurde ein Rechtecksignal mit der Amplitude 2 und der Frequenz f = 1Hz gewählt.

Das Ergebnis der Berechnung des Regelkreises für die Regelgröße ist in Bild (5) dargestellt. Die dazugehörige Stellgröße ist in Bild (6) zu sehen.

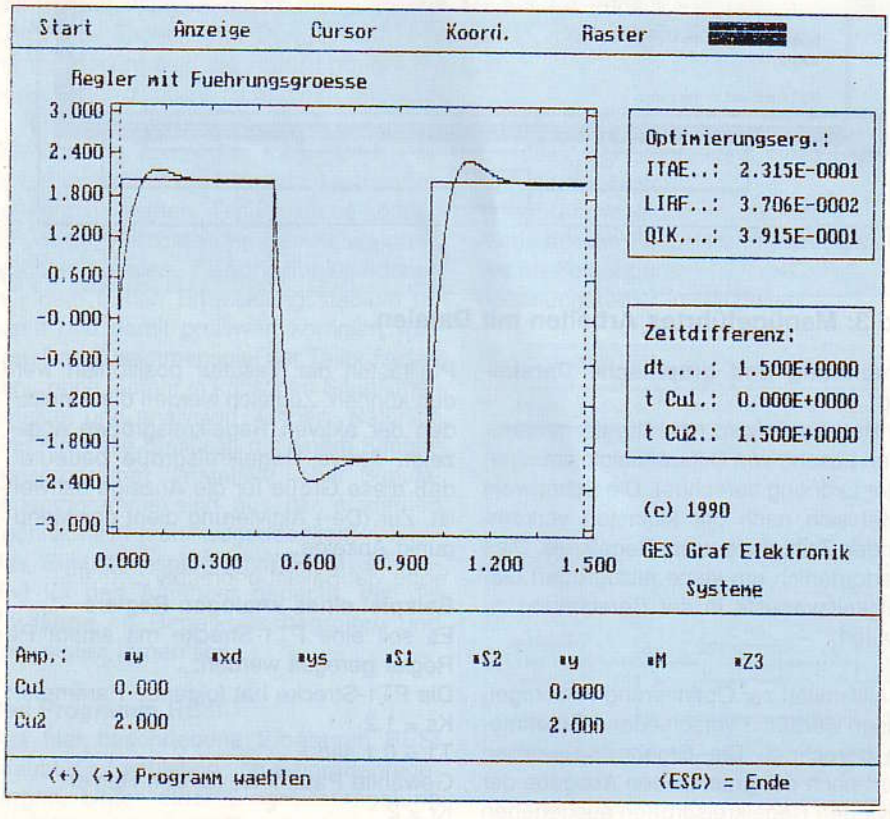


Bild 5: Regelgröße eines konkreten Regelkreises

Beispiel zur digitalen Regelung
(interessant im Zusammenhang mit de
Grundlagen der digtalen Filterung)
Der analoge Regler aus dem obigen Be
spiel soll durch einen digitalen Rechn
nachgebildet werden.

Ein PI-Regler läßt sich mathematisch m
folgender Formel beschreiben:

$$y(t) = Kr[x(t) + 1/Tn \int x(T)dT] \quad (1)$$

mit
y(t) = Stellgröße —>y(k)
x(t) = Eingangsgröße —>x(k)
Kr = Verstärkungsfaktor —>Kr
Tn = Nachstellzeit —>Tn

für y(k) gilt:
y(k) = y(t) zum Zeitpunkt k*To und
x(k) = x(t) zum Zeitpunkt k*Ta
Ta ist die Abtastfrequenz

Durch die Verwendung kleiner Abtastze
ten Ta (Ta << Tn) kann Gleichung (1) durc
diskretisieren in eine Differenzengleichun
umgewandelt werden:

$$y(k) = Kr[x(k) + Ta/Tn * x(k)] \quad (2)$$

Diese Berechnung von y(k) wird als Ste
lungsalgorithmus bezeichnet, da zum Zei
punkt der Berechnung alle x(k) bekann
sein müssen.
Dieser Algorithmus eignet sich aus 2 Grün
den nicht für eine Programmierung au
einem Prozessrechner:

1. Es müssen vom Start ab alle x(k) im Speicher des Rechners abgelegt werden.
2. Bei größeren k dauert die Berechnung von y(k) auch bei einer sehr schnellen CPU unvertretbar lange.

Abhilfe schafft man, indem der Algorith
mus aus Gleichung (2) in einen rekursive
Algorithmus überführt wird.
Beim rekursiven Algorithmus wird de
Stellwert mit Hilfe des vorhergehende
Stellwerts y(k-1) berechnet. Die gesamt
'Vorgeschichte' befindet sich implizit in de
Stellgröße y(k-1)!

Für y(k-1) gilt:

$$y(k-1) = Kr[x(k-1) + Ta/Tn * x(k-1)] \quad (3)$$

Durch Subtraktion der Gleichung (3) vo
der Gleichung (2) erhält man:

$$y(k) - y(k-1) = Kr[x(k) + Ta/Tn * x(k) - x(k-1)] \quad (4)$$

Dieser Algorithmus kann recht einfach au
einem Rechner programmiert werden.
Dieser Regler soll nun mit Hilfe des Pro

gramms RESI getestet werden. Dazu wird der rekursive Algorithmus angewählt. Das Programm erwartet nun eine Eingabe der Faktoren q0..q5. Die Bezeichnung qn wird in der digitalen Regelungstechnik verwendet. Es sollen nun die Faktoren qn aus der Gleichung (4) berechnet werden.

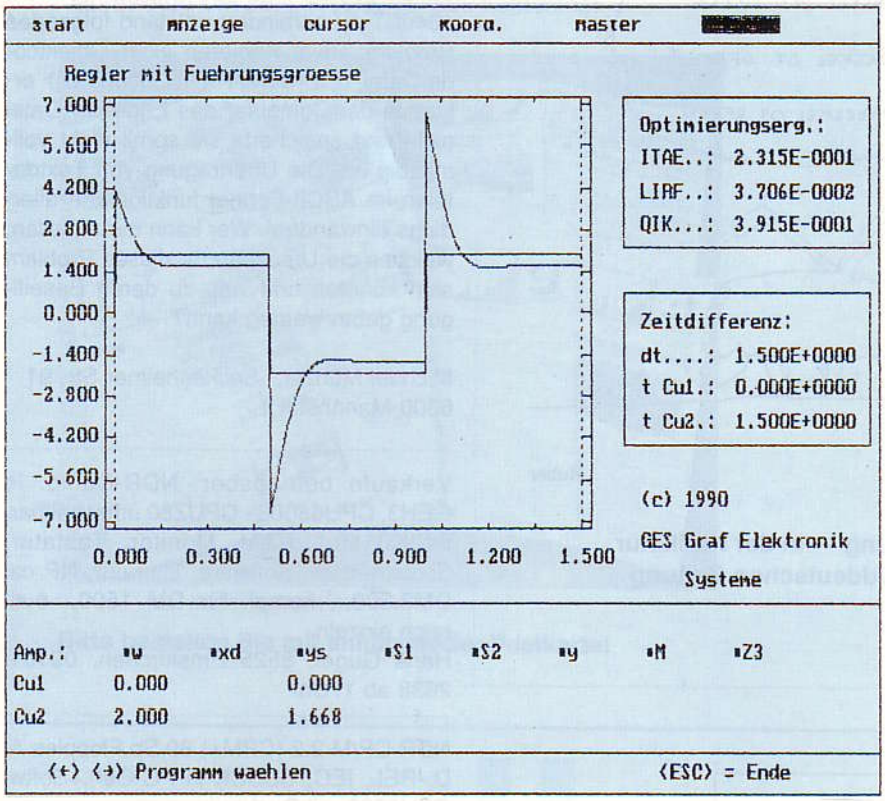


Bild 6: Stellgröße, wie sie der Regel ausgibt

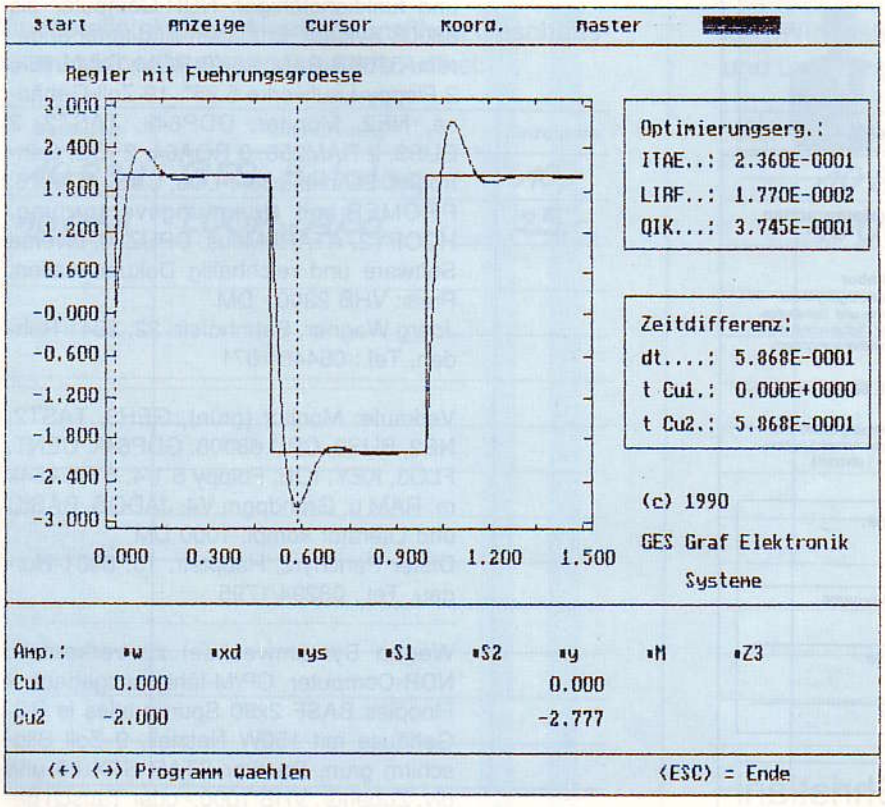


Bild 7: Ergebnis eines Regelkreises mit digitalem Regler

$$y(k) = y(k-1) + Kr(1+Ta/Tn)x(k) - Kr\,x(k-1) \quad (5)$$

$$\Longrightarrow \quad q0 = Kr(1+Ta/Tn)$$
$$q1 = -Kr$$

Damit erhät man:

$$y(k) = y(k-1) + q0\,x(k) + q1\,x(k-1) \quad (6)$$

Es werden also nur die beiden Faktoren q0 und q1 benötigt, denn es handelt sich um eine Differenzengleichung erster Ordnung. Bei einer Differenzengleichung zweiter Ordnung würden die Faktoren q0, q1 und q2 benötigt usw.

Das bedeutet, daß sich für den Fall kleiner Abtastzeiten die Reglerparameter q0 und q1 aus den bekannten Reglerparamtern Kr und Tn des analogen Reglers berechnen lassen (siehe 4.1):

$$Ta = Tn / 10 = 0.0025$$

$$q0 = Kr(1+Ta/Tn) = 2(1+0.0025/0.025) = 2.2$$
$$q1 = -2;$$

Die Strecke sowie die Anregungsfunktion haben die gleiche Einstellung wie im obigen, analogen Beispiel.

Das Ergebnis der Führungsgrößenantwort des Regelkreises ist in Bild (7) zu sehen.

Ein Vergleich mit dem Ergebnis des analogen Beispiels zeigt fast das gleiche Regelverhalten. Das bedeutet, daß analoge Regler sehr einfach durch digitale Regler ersetzt werden können.

Zusammenfassung

Das Programm RESI erlaubt durch seine freie Konfigurierbarkeit des Regelkreises alle denkbaren Regelkreise zusammenzustellen. Durch die Darstellung aller Regelkreisgrößen (in der Praxis oft nicht möglich und wenn dann nur mit enormem Aufwand) ist es vor allem für den Einsteiger in die Regelungstechnik ein wertvolles Lernmittel. Eine Optimierung der Regler ist anhand der Berechnung von 3 Gütekriterien einfach durchzuführen. Durch die Implementierung der digitalen Regler wird modernste Technik transparent dargestellt und ohne viel Ballast begreifbar gemacht.

Die integrierte Hardcopy-Funktion erlaubt eine Dokumentation der erarbeiteten Ergebnisse.

Durch die fehlertolerante Editierung und selbsterklärende Bedienoberfläche wird das Programm RESI auch für den Computer-Laien nutzbar.

ELINT – elektronische Mustererkennung – in der Karikatur ein Cartoon aus der Süddeutschen Zeitung

Leserbrief / Kleinanzeigen

Bei dem Versuch, den NDR-Computer (läuft mit CPU68000 unter CP/M68k) über die serielle Schnittstelle mit einem ATARI 1040ST zu verbinden entstand folgendes Problem: Beim Kopieren einer Objektcode-Datei (PIP-Befehl mit Option [O]) erkannte der Computer das Ende der Datei nicht und speicherte sie somit nicht vollständig ab. Die Übertragung von Textdateien im ASCII-Format funktionierte allerdings einwandfrei. Wer kann mir mitteilen, welches die Ursachen für dieses Problem sein könnten und was zu deren Beseitigung getan werden kann?

Michael Münzer, Seckenheimer Str. 91
6800 Mannheim 1

Verkaufe betriebsber. NDR-Comp. in GEH3, CPU68008 - CPUZ80 umschaltbar, 640KB stat. RAM, Monitor, Tastatur, Schaltnetzteil, Software, Literatur; NP ca. DM3.300,-, kompl. für DM 1500,- evtl. auch einzeln.
Hans Gugel, 8525 Emskirchen, 09101/2638 ab 17Uhr

NDR CP/M 2.2 (CPM+) 80 Sp.Floppies A/D, REL, IEC, CLOCK in PC-Geh.+Softw. CPU8088 mit Buskopp.
Tel.abends 02158/5517

Gelegenheit: Verkaufe voll ausgebauten und funktionsfähigen NDR-Computer als 68008-System. Enthalten sind unter anderem: 576KB RAM, 88KB ROM, CP/M-68k, 2 Floppy-Laufwerke 5.25", 19-Zoll-Gehäuse, NE2, Monitor, GDP64k, TAST2, 2 BUS3, 2 RAM256, 3 ROA64, 2 IOE, Centronics-Schnittstelle, FLO3, CAS, AD8x16, PROMER mit Spannungsversorgung, HCOPY2, ATARI-Maus, CPUZ80, diverse Software und reichhaltig Dokumentation. Preis: VHB 2300,- DM.
Joerg Wagner, Bahnhofstr. 22, 2841 Rehden, Tel.: 05446/1871

Verkaufe: Monitor (grün), GEH3, TAST2, NE2, BUS3, CPU 68008, GDP64k, CENT, FLO3, KEY, IOE, Folppy 5 1/4, 5 ROA64k m. RAM u. Grundpgm.V4, JADOS, BASIC und Literatur kompl. 1000 DM
Dieter Panchyrz, Hauptstr. 15, 8901 Horgau, Tel.: 08294/1795

Wegen Systemwechsel zu verkaufen: NDR-Computer, CP/M-fähig ausgebaut, 2 Floppies BASF 2x80 Spuren alles in PC-Gehäuse mit 150W Netzteil, 9 Zoll Bildschirm grün, Drucker STAR STX 80 und div. Zubehör. VHB 1000,- oder Tausch gegen Farbmonitor für PC Tel.: 06134/65290

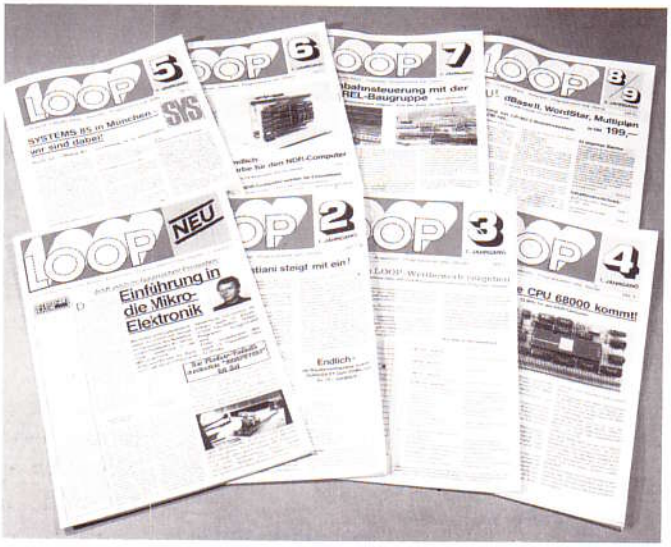
Auch nach dem Kauf:

Die Computerzeitschrift *LOOP* ist die Brücke zum Kunden – Programme, Infos, Tips + Tricks!

Jahres-Abo DM 25,–, Probeexemplar kostenlos!

Umfassend informiert Sie unser Katalog.

Schutzgebühr: DM 10,– incl. MWSt.



Bitte bestellen Sie mit anhängender Postkarte!

BESTELLKARTE

Ich / Wir bestelle(n) unter Anerkennung Ihrer Geschäfts- und Lieferungsbedingungen folgende Artikel:

Stück	Bestell-Nr.	Bezeichnung	Einzelpreis
	10 834	GES-Katalog	10,-
	10061	LOOP-Abo	25,-

Adresse (umseitig) nicht vergessen!

Datum

Unterschrift
Bei Minderjährigen die des gesetzl. Vertreters

BESTELLKARTE

Ich / Wir bestelle(n) unter Anerkennung Ihrer Geschäfts- und Lieferungsbedingungen folgende Artikel:

Stück	Bestell-Nr.	Bezeichnung	Einzelpreis
	10 834	GES-Katalog	10,-
	10061	LOOP-Abo	25,-

Adresse (umseitig) nicht vergessen!

Datum

Unterschrift
Bei Minderjährigen die des gesetzl. Vertreters

Neue Produkte - Neue Preise

IBM-PC-Kompatible

Best.-Nr.		Preis DM
11503	PROMER2IBMF Promer-Fertigg. für PC-Bus, incl. Software	428,-
11502	PROMER2IBMB Promer-Bausatz, incl. Software	378,-
11504	PROMER2IBMP Platinen + Soft- ware	148,-
11352	PROFILOGIBM38 V5.0-01 Profiversion des LogSim 3 1/2", jetzt mit Muti I/O-Anpassung!	369,-
11353	PROFILOGIBM58 V5.0-01 Profiversion des LogSim 5 1/4", jetzt mit Multi I/O-Anpassung	369,-

NDR-und ECB-Systeme

Best.-Nr.		Preis DM
11295	PROMER2F Promer-Fertigg. f. NDR-BUS	348,-
11436	PROMER2ECBF Promer-Fertigg. f. ECB-Systeme	440,-
11294	PROMER2B Promerbausatz f. NDR-Systeme	298,-
11435	PROMER2ECBB Promerbausatz f. ECB-Systeme	307,80
11296	PROMER2P Platinen zum Promer2	59,-
11345	MULTIOF Multi I/O Fertiggerät für PC-und NDR-Systeme	398,-
11344	MULTI IOEXTF Multi I/O Extern für einfache Versuchsauf- bauten	289,-

Bitte Porto nicht vergessen

ANTWORT

GRAF computer

Graf Elektronik Systeme GmbH

Postfach 1610

8960 Kempten

Anschrift: _____

_____

_____ — _____

Lieferform: ☐ Nachnahme ☐ Vorkasse

☐ Bankeinzug

Bankeinzug: Hiermit ermächtige ich die Firma GES GmbH, den Rechnungsbetrag für die auf dieser Karte angegebenen Bestellungen von meinem Konto:

BLZ _____ Konto-Nr. _____

Bank: _____

abzubuchen. Falls mein Konto die erforderliche Deckung nicht aufweist, besteht seitens des kontoführenden Kreditinstitutes keine Verpflichtung zur Einlösung.

Datum _____ Unterschrift _____

Bitte Porto nicht vergessen

ANTWORT

GRAF computer

Graf Elektronik Systeme GmbH

Postfach 1610

8960 Kempten

Anschrift: _____

_____

_____ — _____

Lieferform: ☐ Nachnahme ☐ Vorkasse

☐ Bankeinzug

Bankeinzug: Hiermit ermächtige ich die Firma GES GmbH, den Rechnungsbetrag für die auf dieser Karte angegebenen Bestellungen von meinem Konto:

BLZ _____ Konto-Nr. _____

Bank: _____

abzubuchen. Falls mein Konto die erforderliche Deckung nicht aufweist, besteht seitens des kontoführenden Kreditinstitutes keine Verpflichtung zur Einlösung.

Datum _____ Unterschrift _____